AFRL-IF-RS-TR-2007-104
Final Technical Report
April 2007

ENHANCEMENT OF THE COMPUTATIONAL EFFICIENCY OF MEMBRANE COMPUTING MODELS

Digendra K. Das

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report was cleared for public release by the Air Force Research Laboratory Rome Research Site Public Affairs Office and is available to the general public, including foreign nationals. Copies may be obtained from the Defense Technical Information Center (DTIC) (http://www.dtic.mil).

AFRL-IF-RS-TR-2007-104 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE DIRECTOR:

/s/ /s/

THOMAS E. RENZ
Work Unit Manager

JAMES A. COLLINS, Deputy Chief
Advanced Computing Division
Information Directorate

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
APR 2007	Final	Oct 05 – Jan 07

4. TITLE AND SUBTITLE

ENHANCEMENT OF THE COMPUTATIONAL EFFICIENCY OF MEMBRANE COMPUTING MODELS

5a. CONTRACT NUMBER

5b. GRANT NUMBER
FA8750-05-2-0043

5c. PROGRAM ELEMENT NUMBER
61101E

6. AUTHOR(S)

Digen Das

5d. PROJECT NUMBER
NBGQ

5e. TASK NUMBER
10

5f. WORK UNIT NUMBER
03

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Digendra K. Das
10056 Mallory Rd
Sauquoit NY 13456-2510

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

AFRL/IFTC
525 Brooks Rd
Rome NY 13441-4505

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSORING/MONITORING AGENCY REPORT NUMBER
AFRL-IF-RS-TR-2007-104

12. DISTRIBUTION AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED. PA# 07-157

13. SUPPLEMENTARY NOTES

14. ABSTRACT
Part I: The researchers developed a comprehensive transitional P-system Membrane Computing model. Membrane computing consists of cell-like membranes placed inside a unique "skin" membrane. In regions delimited by a membrane structure, cells are placed in multisets of objects which evolve according to evolution rules associated with the regions. The researchers considered multi-sets of objects, usually multi-sets of symbols, objects and a set of evolution rules, which are placed inside regions delimited by membranes. The evolution between system configurations is completed non-deterministically by applying rules synchronously in a maximum parallel manner.

Part II: The researchers applied the P system constructs described in Part I to provide end-to-end secure mobile ad hoc networks as Distributed P systems consisting of migrating membrane agents. Membrane networks were modeled, with migration components and a guardian membrane that regulates interactions between the processing component and the external environment. Membranes act as filters that control access to the associated site and rely on the established notion of trust between sites. The researchers developed steps necessary to control the actions of incoming agents and encompass complex policies, wherein the number of actions a membrane agent is allowed to perform and the order of actions are prioritized.

15. SUBJECT TERMS
Membrane Computing, P Systems, Bioinspired Computing

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Thomas Renz
U	U	U	UL	56	19b. TELEPHONE NUMBER (*Include area code*)

Abstract

Part I – P system model development: In Part I the researchers developed a comprehensive transitional P-system Membrane Computing model. Membrane computing consists of cell-like membranes placed inside a unique "skin" membrane. In regions delimited by a membrane structure, cells are placed in multisets of objects which evolve according to evolution rules associated with the regions. The researchers considered multi-sets of objects, usually multi-sets of symbols, objects and a set of evolution rules, which are placed inside regions delimited by membranes. The evolution between system configurations is completed non-deterministically by applying rules synchronously in a maximum parallel manner.

Part II - Application of P system models. In Part II the researchers applied the P system constructs described in Part I to provide end-to-end secure mobile ad hoc networks as Distributed P systems consisting of migrating membrane agents. Membrane networks were modeled, with migration components and a guardian membrane that regulates interactions between the processing component and the external environment. Membranes act as filters that control access to the associated site and rely on the established notion of trust between sites. The researchers developed steps necessary to control the actions of incoming agents and encompass complex policies, wherein the number of actions a membrane agent is allowed to perform and the order of actions are prioritized.

Table of Contents

Part I – P System Model Development

Part II – Application of P System Models

Part I – P System Model Development

1.0 Summary

Part I of this research project consists of the scientific and mathematical constructs and algorithms of the Membrane Computing models. The work consisted of the following four tasks:

Task 1.1 Development of the Scientific and mathematical aspects of the Membrane Computing model and the relevant algorithms incorporating the Membrane Division technique.

Task 1.2. Development of the Scientific and mathematical aspects of the Membrane Computing model and the relevant algorithms incorporating the Membrane Creation technique.

Task 1.3. Development of the Scientific and mathematical aspects of the Membrane Computing model and the relevant algorithms incorporating the Membrane String Replication technique.

Task 1.4. Design of the preliminary experimental setups on Membrane Structure to support the verification of the Membrane Computing Model developed in Tasks 1.1-1.3.

2.0 Introduction

P systems are a class of distributed parallel computing models inspired from the way that live cells process chemical compounds, energy and information. It is highly parallel and is based on the notion of a membrane structure. Such a structure consists of several cell-like membranes recurrently placed inside a unique skin membrane. In the regions delimited by a membrane structure are placed multisets of objects, which evolve according to evolution rules associated with the regions. These objects placed in the regions delimited by the membranes, can be transformed into other objects and can pass through a membrane. The researchers considered two basic classes of P systems, symbol-objects and string-objects. The usefulness of these models was tested by solving the Hamiltonian Path Problem (HPP) for Undirected Graphs. The relevant algorithms for the above mentioned problem were subsequently designed and incorporated into the development of a P system simulator [12, 13].

3.0 Methods, Assumptions and Procedures

3.1 Task 1.1 Development of a P system model with Active Membrane using Membrane Division Technique [1,2,3,4,10]

The P system with active membranes is a construct:

$$\prod = (V, T, H, \mu, w_1, w_2, \ldots\ldots\ldots\ldots..., w_m, R)$$

Where:

i) $m \geq 1$ (the initial state of the system)
ii) V is an alphabet
iii) $T \subseteq V$ (the terminal alphabet
iv) H is a finite set of labels for membranes

v) μ is a membrane structure, consisting of m membranes, labeled with elements of H; all membranes in μ are supposed to be of neutral charge

vi) $w_1, w_2, \ldots\ldots\ldots\ldots, w_m$ are strings over V describing the multisets of objects placed in the m regions of μ

vii) R is a finite set of developmental rules, of the following forms:

Object evolution rules: The object evolution rules are associated with membranes and depend on the label and the charge of the membranes, but do not directly involve the membranes. The membranes are neither taking part in the application of the rules nor are they modified by the object evolution rules.

$$\left[_h \; a \rightarrow v \; \right]^{\alpha}_h$$

for $h \in H$, $\alpha \in \{+, -, 0\}$, $a \in V$, $v \in V^*$

Communication rules: An object is introduced in the membrane, may be modified during the process, also the polarization of the membrane can be modified, but not its label.

$$a\left[_h \; \right]^{\alpha 1}_h \rightarrow \left[_h \; b \; \right]^{\alpha 2}_h$$

for $h \in H$, $\alpha_1, \alpha_2 \in \{+, -. 0\}$, $a, b \in V$

Communication rules: An object is sent out of the membrane and may be modified during this process; also the polarization of the membrane can be modified, but not its label.

$$\left[_h \; a \; \right]^{\alpha 1}_h \rightarrow \left[_h \; \right]^{\alpha 2}_h b$$

for $h \in H$, $\alpha_1, \alpha_2 \in \{+, -, 0\}$, $a, b \in V$

Dissolving rules: In reaction with an object, a membrane can be dissolved, while the object specified in the rule can be modified.

$$\left[_h \; a \; \right]^{\alpha}_h \rightarrow b$$

for $h \in H$, $\alpha \in \{+, -, 0\}$, $a, b \in V$

Division rules for Elementary membranes: The membrane is divided into two membranes with the same label, may be of different polarizations in reaction with an object. The object specified in the rule is replaced in the two new membranes by possibly new objects.

$$\left[_h \; a \; \right]^{\alpha 1}_h \rightarrow \left[_h \; b \; \right]^{\alpha 2}_h \left[_h \; c \; \right]^{\alpha 3}_h$$

for $h \in H$, $\alpha_1, \alpha_2, \alpha_3 \in \{+, -. 0\}$, $a, b, c \in V$

Division rules for non-elementary membranes: If a membrane contains two immediately lower membranes of opposite polarizations, + and -; then the membranes of opposite polarizations are separated in the two new membranes, but their polarization can change. All membranes of opposite polarizations are separated by this rule.

$$\left[_{h_0} \left[_{h_1} \; \right]^{\alpha_1}_{h_1} \ldots\ldots \left[_{h_k} \; \right]^{\alpha_1}_{h_k} \left[_{h_{k+1}} \; \right]^{\alpha_2}_{h_{k+1}} \ldots\ldots \left[_{h_n} \; \right]^{\alpha_2}_{h_n} \right]^{\alpha_0}_{h_0} \rightarrow$$

$$\left[_{h_0} \left[_{h_1} \; \right]^{\alpha_3}_{h_1} \ldots\ldots \left[_{h_k} \; \right]^{\alpha_3}_{h_k} \right]^{\alpha_5}_{h_0} \left[_{h_0} \left[_{h_{k+1}} \; \right]^{\alpha_4}_{h_{k+1}} \ldots\ldots \left[_{h_n} \; \right]^{\alpha_4}_{h_n} \right]^{\alpha_6}_{h_0}$$

for $k \geq 1$, $n > k$, $h_i \in H$, $0 \leq i \leq n$ and $\alpha_0, \alpha_1, \ldots\ldots, \alpha_6 \in \{+, -, 0\}$ with $\{ \alpha_1, \alpha_2 \} = \{+, -\}$

These rules are applied according to the following principles:

1. All rules are applied in parallel.
2. If a membrane is dissolved, then all the objects in its region are left free in the surrounding region. The skin membrane is never dissolved.
3. All objects and membranes not specified in a rule and which do not evolve pass unchanged to the next step. For instance, if a membrane with a label h is divided by a division rule for elementary membranes, which involves an object a, then all other objects in membrane h, which do not evolve, are introduced in each of the two resulting membranes h. Similarly, when dividing a membrane h by means of a division rule for non-elementary membranes, the neutral membranes are reproduced in each of the two new membranes with the label h, unchanged if no rule is applied to them (in particular, the contents of these neutral membranes are reproduced unchanged in these copies, providing that no rule applied to their objects.
4. If at the same time a membrane h is divided by a division rule for an elementary membrane and there are objects in this membrane, which evolve by means of object evolution rules. Then, in the new copies of the membrane, we introduce the result of the evolution. First, the object evolution rules are used; changing the objects, then the division is produced, so that in the two new membranes with label h we introduce the copies of the changed objects. This process takes only one step. The same assertions apply to the division of non-elementary membranes. First, the rules of the innermost region are applied and then level by level until the region of the skin membrane.
5. The rules associated with a membrane h are used for all copies of this membrane, irrespective of whether or not the membrane is an initial one or it is obtained by division. At one step, a membrane can be subject to only one type of rule.
6. The skin membrane can never divide. As any other membrane, the skin membrane can be electrically charged.

3.2 Task 1.2. Development of a P system model with Active Membrane using Membrane Creation Technique [5,6,7,11]

A P system with Membrane Creation of degree (m, n), $n \geq m \geq 1$, is a construct:

$$\Pi = (V, T, C, \mu, w_0, w_1 \ldots\ldots\ldots\ldots w_{(m-1)}, R_0, R_1, \ldots\ldots, R_{(n-1)}),$$

Where:

1. V is an alphabet; it consists of both productive and non-productive objects;
2. $T \subseteq V$, is the output alphabet;
3. $C \cap V \neq \varphi$, is the set of catalysts;
4. μ is a membrane structure consisting of m membranes, with the membranes and the regions labeled in a one- to-one manner with elements in a given set; here the labels are used as 0 (for the skin membrane), 1, ……, (m-1);
5. w_i , $0 \leq i \leq (m-1)$, are multi-sets of objects over V associated with the regions 0, 1,…., (m-1) of μ;
6. R_i , , $0 \leq i \leq (n-1)$, are finite set of evolution rules over V.

An evolution rule is of two types:

(a) If a is a single non-productive object, then the evolution rule is in the form $a \rightarrow v$ or $c\ a \rightarrow c\ v$, where $c \in C$, $a \in (V - C)$, $v = v'$ or $v = v'\delta$ or $v = v'\tau$, where v' is a multi-set of objects over $((V\text{-}C) \times (here, out)) \cup ((V\text{-}C) \times (in_j \mid 1 \leq j \leq (n-1)))$, and δ, τ are special symbols not in V.

(b) If a is a single productive object, then the evolution rule is in the form $a \rightarrow [_i\ v\ ]_i$ or $c\ a \rightarrow c\ [_i\ v\ ]_i$, where $c \in C$, $a \in (V - C)$, v is a multi-set of objects over (V-C). The rule of the form $a \rightarrow [_i\ v\ ]_i$ means that the object identified by a is transformed into the object identified by v, surrounded by a new membrane having the label i. No rule of the form $a \rightarrow [_0\ v\ ]_0$ can appear in any set R_i. During a computation the number of membranes can increase or decrease.

The membrane structure and the multi-sets in Π constitute the initial configuration of the system. One can pass from a configuration to another one by using the evolution rules. This is done in parallel: all objects, from all membranes, which can be the subject of local evolution rules, should evolve simultaneously. A rule can be used only if there are objects which are free at the moment one checks its applicability.

The application of a rule $c\ a \rightarrow c\ v$ in a region containing a multi-set w means to remove a copy of the object a in the presence of c (catalyst), provided that such copies exist, then follow the prescription given by v: If an object appears in v in the form (a, here), then it remains in the same region; if it appears in the form (a, out), then a copy of the object a will be introduced in the region of the membrane placed outside the region of the rule $c\ a \rightarrow c\ v$; if it appears in the form (a, in_i), then a copy of a is introduced in the membrane with index i, if such a membrane exist inside the current membrane, otherwise the rule can not be applied. If the special symbol τ appears, then the thickness of the membrane which delimits the region where we work is increased by 1. Initially, all membranes have the thickness 1. If a rule in a membrane of thickness 1 introduces the symbol τ, then the thickness of the membrane becomes 2. A membrane of thickness 2 does not become thicker by using further rules which introduce the symbol τ, but no object can enter or exit it. If a rule which introduces the symbol δ is used in a membrane (including the skin membrane) of thickness 1, then the membrane is dissolved; if the membrane had thickness 2, then it returns to thickness 1. Whenever the skin membrane is dissolved, the whole membrane system will be destroyed. If at the same step one uses rules which introduce both δ and τ in the same membrane, then the membrane does not change its thickness. No object can be communicated through a membrane of thickness two; hence rules which introduce commands out and in_j requesting such communications cannot be used. However, the communication has priority over changing the thickness. If at the same step an object should be communicated and a rule introduces the action τ, then the object is communicated and "after that" the membrane changes the thickness. When applying a rule $a \rightarrow [_i\ v\ ]_i$ in a region j, a copy of *a* is removed and a membrane with the label i is created, containing the multi-set v, inside the region of membrane j. We can never create a skin membrane. Similarly, when applying a rule $c\ a \rightarrow c\ [_i\ v\ ]_i$ in a

region j, a copy of a, in presence of c (catalyst), is removed and a membrane with the label i is created, containing the multi set v, inside the region of membrane j.

A sequence of transitions between configurations of a given P system Π is called a Computation with respect to Π. A computation is successful if and only if it halts, that is, there is no rule applicable to the objects present in the last configuration. The result of a successful computation is $\psi_T(w)$ as denoted by Ps(Π) (from "Parikh set") and we say that it is generated by Π. (Note that we take into account only the objects from T).

The model was applied to solve the following problem: Hamiltonian Path Problem (HPP) for Undirected Graphs.

3.3 Task 1.3. Development of P systems with String Replication Technique [8,9]

In P systems, with objects described by strings, we work with multisets of strings and consider the result of a computation as the number of strings in a given output membrane. The strings (also called worms) are processed by replication, splitting, mutation and recombination. P systems of this type can generate all recursively enumerable sets of numbers. The string processing operations are described below.

String Processing Operations

We denote by V^*, the free monoid generated by alphabet V under the operation of concatenation; the empty string is denoted by λ, $V^+ = V^* - \{\lambda\}$ is the set of non-empty strings over V, and $|x|$ is the length of $x \in V^*$.
We consider the following operations on strings over alphabet V:

1. Replication: If $a \in V$ *and* $u_1, u_2 \in V^+$, *then r:* $a \rightarrow u_1 \,||\, u_2$ is called a replication rule.

For strings $w_1, w_2, w_3 \in V^+$ we write:
$w_1 \Rightarrow_r (w_2, w_3)$; w_1 is replicated with respect to rule r
if $w_1 = x_1 a x_2$, $w_2 = x_1 u_1 x_2$, $w_3 = x_1 u_2 x_2$, for some $x_1, x_2 \in V^*$

2. Splitting: If $a \in V$ *and* $u_1, u_2 \in V^+$, *then r:* $a \rightarrow u_1 \mid u_2$ is called a splitting rule.

For strings $w_1, w_2, w_3 \in V^+$ we write:
$w_1 \Rightarrow_r (w_2, w_3)$; w_1 is split with respect to rule r
if $w_1 = x_1 a x_2$, $w_2 = x_1 u_1$, $w_3 = u_2 x_2$, for some $x_1, x_2 \in V^*$

3. Mutation: A mutation rule is a context-free rewriting rule, $a \rightarrow u$, over V.

For strings $w_1, w_2 \in V^+$ we write:
$w_1 \Rightarrow_r w_2$
if $w_1 = x_1 a x_2$, $w_2 = x_1 u x_2$ for some $x_1, x_2 \in V^*$

4. Recombination: Consider a string $z \in V^+$ (as a crossing-over block) and four strings $w_1, w_2, w_3, w_4 \in V^+$. We write:

$(w_1, w_2) \mid\!-_z (w_3, w_4)$
if $w_1 = x_1 z x_2$, $w_2 = y_1 z y_2$, $w_3 = x_1 z y_2$ and $w_4 = y_1 z x_2$ for some $x_1, x_2, y_1, y_2 \in V^*$

Note that replication and splitting increase the number of strings while mutation and recombination do not.

P systems with Worm-objects.

A P system with worm-objects is a construct

$$\prod = (V, \mu, A_1, \ldots\ldots, A_m, (R_1, S_1, M_1, C_1), \ldots\ldots, (R_m, S_m, M_m, C_m), i_0),$$

where:

- V is an alphabet;
- μ is a membrane structure of degree m (with m membranes);
- $A_1, \ldots\ldots, A_m$ are multisets of finite support over V^*, associated with the regions of μ (the initial populations of worms)
- for each $1 \leq i \leq m$, R_i, S_i, M_i, C_i are finite sets of replication rules, splitting rules, mutation rules, and crossing over blocks, respectively, given in the following forms:

  a. replication rules: $(a \rightarrow u_1 \,||\, u_2; tar_1, tar_2)$ or $(a \rightarrow u_1 \,||\, u_2; tar_1, tar_2)\delta$, for $tar_1, tar_2 \in \{here, out\} \cup \{in_j \mid 1 \leq j \leq m\}$;

  b. splitting rules: $(a \rightarrow u_1 \mid u_2; tar_1, tar_2)$ or $(a \rightarrow u_1 \mid u_2; tar_1, tar_2)\delta$, for $tar_1, tar_2 \in \{here, out\} \cup \{in_j \mid 1 \leq j \leq m\}$;

  c. mutation rules: $(a \rightarrow u; tar)$ or $(a \rightarrow u; tar)\delta$, for $tar \in \{here, out\} \cup \{in_j \mid 1 \leq j \leq m\}$;

  d. crossing-over blocks: $(z; tar_1, tar_2)$ or $(z; tar_1, tar_2)\delta$, for $tar_1, tar_2 \in \{here, out\} \cup \{in_j \mid 1 \leq j \leq m\}$;

- $i_0 \in \{1, 2, \ldots\ldots, m\}$ specifies the *output membrane* of the system

The $(m+1)$-tuple $(\mu, A_1, \ldots\ldots, A_m)$ constitutes the initial configuration of the system. By applying the operations defined by the components (R_i, S_i, M_i, C_i), $1 \leq i \leq m$, we can pass from one configuration to another. Following principles are followed:

1. The operations are applied in a maximally parallel manner, that is, all strings which can be processed by means of rules in a particular region, are processed in each time unit.
2. The rules to be used and the copies to be processed are chosen in a non-deterministic manner.
3. A string is processed by only one operation. We cannot apply two mutation rules, or a mutation rule and a replication rule, to the same string.
4. The strings resulting from an operation (two, in the case of replication, splitting, and recombination, and one, in the case of mutation) are transferred to the region specified by the target indications associated with rules:

here means that the string remains in the same region where the rule has been applied, *out* means that the string is sent out of that region (a string can also leave the skin membrane), in_j means that the string is sent to membrane j, providing that this membrane is adjacent to the region where the rule is applied.

5. When a rule is applied which also contains the symbol δ, the current membrane is dissolved; all its strings are left free in the membrane directly above it, while its rules and crossing-over blocks are lost. The skin membrane is never dissolved. The application of rules is supposed to take place from bottom-up.

A sequence of transitions, starting from the initial configurations, is called a computation. A computation is complete if it halts; that is no further rule can be applied to strings in the last configuration. If at the end of a complete computation, membrane i_0 is present in the system (it was not dissolved during the computation) and it is an elementary one, then the number of strings from region i_0 is the result of the computation. A non-halting computation provides no output.

3.4 Task 1.4. Design of the preliminary experimental setups on Membrane Structure to support the verification of the Membrane Computing Model.

Active transport of molecules is a part of numerous biochemical reactions which are important for various cellular events in a living organism. Among different biochemical processes, a number of studies have been published on neurotransmitter transporters. During neural activity, neurotransmitters are released into the synaptic clefts where they act on various receptors. These transmitter molecules are recycled by another group of molecules, known as neurotransmitter transporters. A lot of progress has recently been made on the study of neurotransmitter transporters due to progress in the cloning of different genes that encode different proteins.

These transporter molecules are located in the plasma membranes of the nerve cells (neurons). Transporters of monoamines, including many amino acids created by these transporters, are sodium and chloride ion dependent. Study of these transporters is important not only for the progress of science, but also has great medical significance. For example, molecular inhibition of dopamine (e.g., a neurotransmitter) transporter has been linked to the euphoria and the reinforcing properties of psycho stimulants like cocaine and amphetamines . Also, major classes of antidepressants act by inhibiting norepinepherine and/or serotonin (other two biologically important neurotransmitters) transporters.

Design of experiments. Using this well studied biological model as our theoretical working system, we can design experiments to show controlled movement of molecules through a biological membrane. In order to do these experiments, we had to have a biological system where these transporters can be expressed in a biological membrane which can be used as the experimental medium to study the active transport of a molecule whose passage through the membrane can be controlled by changing the experimental condition(s).

Transporters responsible for re-uptake of neurotransmitters across plasma membrane of neurons fall into two gene families [14]. Small neurotransmitters including amino acids are transported by proteins belonging to the family called neurotransmitter sodium symporter (NSS). Glutamate is transported by proteins belonging to the family called dicarboxylate amino acid cation symporter (DAACS). Both types of transporters use a transmembrane ion gradient to transport the neurotransmitters across the membrane. For the uphill movement of one molecule of GABA (gamma amino butyric acid), downhill movement of two Na+ ions and one Cl- ion across the membrane is necessary [15]. Serotonin (SERT) transporter couples the inward movement of one Na+ ion one Cl- ion

and one molecule of SERT in cationic form to the outward movement of one K+ or one H+ ion [16]. This coupled inward movement, or symport of Na+ ion and neurotransmitter molecule is a universal feature of the NSS family of transporters. On the other hand, most of the bacterial transporters use H+ ion symport to import metabolites into cells.

Recommended Experimental Designs. Based on these facts, we proposed the following plan to experimentally show the controlled movement of atoms / ions across a bacterial membrane [23].

1. Expression of recombinant protein:

A neurotransmitter whose cDNA (complementary deoxyribonnucleic acid) is available, will be expressed in E. coli, using commercially available QIA, (a registered trademark for a product by the company QUIAGEN), protein expression kit. Today it is a common practice to use E. coli to express a recombinant protein. In brief, gene (cDNA) of a protein of interest is cloned (inserted) into the expression vector (commercially available) leading to the creation of a gene fusion. This is then transformed into E. coli (bacteria). Transformed E. coli are grown in large scale liquid cultures and after the cells reach an appropriate density , they are induced generally with IPTG (isopropyl-Beta-dextro-thiogalactoside) to express the recombinant protein of interest. These transporters are expressed on the plasma membrane of the bacteria.

2. Preparation of Bacterial Membrane Vesicles:

Vesicles are basically the bacterial cells minus the intracellular contents. They are a hollow sphere of bacterial membrane with the bacteria proteins lodged in the membrane. The inner content of the membrane can be manipulated according to the need of the experimental conditions. Right side out vesicles will be prepared by following the procedure already described in [17]. In brief, proteoplasts or spheroplasts are formed when bacterial (E. coli cells after the transporter is expressed) cells are treated with lysozymes and EDTA (ethylene diamine tetra acetic acid), as a result of release of their intracellular contents after they are placed in a hypotonic media. These membranes reseal themselves by an unknown mechanism, yielding a closed empty membrane vesicle which can be precipitated easily.

3. Use of membrane vesicles as the medium for active transport:

These vesicles will be incubated with a radioactive transport substrate in the presence / absence of an energy source to show the regulated movement of the substrate through the membrane by following the procedure described in [18].

4.0 Results and Discussion

As a result of the work completed on above mentioned Tasks 1.1, 1.2 and 1.3 the researchers designed a universal and parallel simulator, referred to as the DasPsimulator, for a special class of transitional P systems. The DasPsimulator is highly configurable and in principle can be used to evolve membrane systems of varying complexity [12, 13].

In task 1.4 the researchers designed the necessary experiments, however the implementation of these experimental designs will be left for future work.

5.0 Conclusions

In Part I of this project, the researchers presented a universal and massively parallel implementation of a special class of P systems. The architecture of the universal membrane allowed the use of the same module anywhere in the hierarchy of a membrane system and independently of the number of rules and objects to be stored within it. The conclusions summarized below were modeled and simulated efficiently using a customized P simulator [13]:

- Membrane Computing provides computational models that abstract from the living cells' structure and function

- Such models have been proven to be computationally powerful (equivalent to a Turing Machine) and efficient (solving NP-Complete problems)

- Membrane Computing defines an abstract framework for reasoning about:
  - distributed architectures
  - communication
  - parallel information processing

- Such features are relevant both for Computer Science (Distributed Computing Models, Multi-Agent Systems) and Networking (Modeling and Simulation of Biological Networks)

6.0 Recommendations for future work

Future work will focus on the development of a hardware based implementation and improvement (in terms of speed and resources used) of the existing architecture using Field Programmable Gate Arrays (FPGA) [35], amalgamated with alternative computational and bio-inspirational metaphors. In addition, the researchers plan to extend the existing design in order to be able to reuse dissolved membranes and in order to apply rules in a fully parallel and nondeterministic manner [36]. In addition, dealing with a larger number of objects would probably require extensive processor capacity [19]. This was not a serious limitation in the current implementation. Furthermore, the researchers also envision extending the current design to a hardware system implementation that could potentially be customized with Application Specific Integrated Circuits (ASIC) and implemented to provide specialized applications [28, 30, 31, 32].

Examples of specialized P-system models:

- Energy-Controlled P-systems
- P-systems with promoters/inhibitors
- P-systems with carriers
- P-systems with mobile membranes
- Tissue P-systems
- Probabilistic P-systems
- P-systems with elementary graph productions
- Parallel Rewriting P-systems [22]

Part II – Application of P System Models

1.0 Summary

Part II of the research project consisted of the following five tasks:

Task 2.1: Membrane types for protocol analysis: This task included developing an extensive array of Distributed P system process calculi and theorems for modeling secure mobile ad hoc networks comprised of membrane agents embedded in Distributed P system models for non-interference and protocol analysis.

Task 2.1: Core Mobility: Here the researchers extended membrane networking techniques for resource management and information flow to cope with security in the absence of centralized control. This included balancing static/dynamic access techniques to maximize accuracy, efficiency and security.

Task 2.3: Membrane Mobility: In this task the researchers investigated the effects of membrane agent mobility on trust level and security clearance for agents and resources and developed a membrane networking system model for dynamically upgrading / downgrading system and network components [86].

Task 2.4: Network Aware Membrane Systems: Here the researchers devised new models of information flow based on causal information and secure Distributed P (DP) systems with focus on using location-aware calculi for securing distributed mobile ad hoc.

Task 2.5: Mobility, Control and Cryptographic primitives: In this task the researchers developed a novel method for the analysis of entity authentication in crypto-protocols.

2.0 Introduction

In this section the researchers summarize the project's activities for applications of the relevant Membrane Computing Models. The researchers began by investigating deterministic models of concurrency [55] (based on non-interleaving actions) as a means to express information-flow [81] and non-interference based networks, by presenting a foundational framework for different approaches [45, 48]. The researchers considered access control policies expressed by deterministic finite automata, with the purpose of focusing on the balance between static and dynamic techniques [65, 66]. The resulting design included a membrane networking model used to enforce discretionary access control. The researchers then developed an expressive Distributed P system based calculus for role-based access control, completed by a membrane networking model used to control access to nodes and information repositories [61]. The researchers also investigated dynamic trust policies, and devised a distributed P system (DP) expressive calculus capable of supporting several trust semantic analyses [46] based on proven bisimulation techniques [62]. This required expanding upon the classic transitional P system rules to specifically address mobile membranes where access is guaranteed either via a (static) certificate vouched by a trusted site, or must be acquired via dynamic membrane security type checking [47]. The secure membrane networking model consists of unique network security levels, selected from a lattice of roles (or certificates), and embedded membrane agents used to obtain access to resources by acquiring appropriate roles [51]. The researchers also introduced a conceptual reputation system based on a count of positive and negative outcomes from membrane interactions, together with a temporal logic to express trust policies and the relative model checking techniques. Next the researchers developed a novel method for the analysis of membrane agent authentication using crypto-protocols [50]. This method was then extended to other types of protocols and attacks, and widened the investigation into message authentication. Several applicable examples were also tested, including the case of multi-protocol systems.

Membrane Types for Protocol Analysis: The main focus here was on constructing bio-inspired membrane networking system models as a vehicle to provide static guarantees of authentication for distributed mobile ad hoc networks [25]. This work builds upon previous research [51] that included membrane computing modeling and algorithm development and proceeded along two conceptually distinct lines of investigation. Initially the researchers developed new methods for the static analysis of entity authentication protocols. Here the main results were used to refine the approach needed to validate protocol participants in a fully compositional way, by deriving an expansion of the transitional P system rules to express protocol narrations; and then validating their results using a number of relevant test cases. The researchers then broadened their approach by extending it to relevant example cases. These include networks where multiple protocols run concurrently, as well as protocols based on both symmetric and asymmetric cryptography and on diverse challenge-response mechanisms; one can now deal with a richer class of network attacks, and verify message authentication properties in addition to membrane entity authentication [67]. Further details and descriptions of the results can be found below (see Section 3.1).

Core Mobility: The main objective here was to undertake a comprehensive background search as a starting point to review the research on distributed mobile ad hoc networks characterized by information flow theory. One possible approach considered was to identify as critical states those where a low-level action depends causally on a high-level one, so that a confined system would be one where critical states are indistinguishable to a low-level user (e.g., untrusted) from authorized ones. A more refined approach was identified that makes use of simulation techniques (see Section 3.2).

Membrane Mobility: This task centered on the notion of policies as specifications of access control capabilities [68], and their enforcement via a mix of static / dynamic techniques. The research contribution consisted of three main lines of inquiry. The first dealt with a general notion of a "guardian membrane agent", used to specify a particular site policy and embedded within the migrating node. If the node is of untrusted origins it is subject to analysis upon arrival. Nodes from trusted locations however can forego the security checking on the grounds that they have been checked somewhere else by a trusted site and were found to be valid. This work required a particular formalism for policy specification, as it was necessary to investigate a series of increasingly complex models for this analysis [47, 48]. In the second approach membranes were grouped into classes and resource access management was controlled by specifying access to named groups rather than by individual nodes. This approach is referred to as Discretionary Access Control [58], (DAC), and allows for flexibility in specification for dynamically changing users as found in the case of mobile ad hoc networks. A third approach undertaken by the researchers included temporal 'sequencing' of information in the access control mechanisms (see Section 3.3).

Network Aware Membrane Systems: Here the researchers adapted policy specification via role-based access control. The focus was on specification to roles rather than membranes or groups of them. Roles identify tasks in the network which any principal could engage in, and the model allows for membranes to dynamically activate and deactivate these roles at run-time. This was then formalized where a static type P system model was used to track the dynamic changes of role and check the compliance of membrane agents' behaviors with respect to policy specifications [39, 40]. Further details and descriptions of the results are described in Section 3.4.

Mobility, Control and Cryptographic primitives: In this task the researchers addressed trust levels primarily focusing on the notion of access control via dynamic trust policies, and arrived at three main sets of results. 1) The researchers expanded upon the transitional P system rules by introducing a Distributed P system calculus of dynamic trust, where trust is not only changeable but is evaluated on demand at the moment when access control is required according to local policies. The researchers derived formulas expressed in a flexible, yet computationally-feasible logic (resembling DataLog), whose evaluation depends on the history of past interactions between network hosts and information repositories (e.g., data bases) and network nodes. 2) The second contribution is provided by the innovative work on role-based access control. Here the researchers designed a set of roles active for a network node that could be interpreted as its clearance

level. Membrane agent data sets vary over time according to the dynamics of nodes, which acquire and release their roles; thereby the researchers identified an important example of dynamic security levels. The researchers also addressed dynamic access policies implemented by deterministic finite automata [29]. 3) A third contribution included the notion of dynamic trust based on reputation mechanisms [81]. A node's reputation is established over time by factoring into account the outcomes of the past interactions. This work required the application of a temporal logic able to express trust properties and temporal evolution, which together with a model-checking algorithm, was used to automatically verify such properties (see Section 3.5).

2.1 Summary of achieved results. The researchers developed a Distributed P system model that was used to validate a number of relevant examples. The researchers developed a bio-inspired secure membrane networking paradigm to provide end-to-end security for distributed mobile ad hoc networks. Next they applied this paradigm to treat relevant examples. The researchers then formalized a set of principles capable of guaranteed security for distributed networking and crypto-protocols. Finally, they verified that all objectives were met, and in some instances even exceeded original expectations. By way of example, their work on core mobility models advanced the groundwork established for the first such known investigation into the relationships between membrane networking models applied to securing distributed mobile ad hoc networks [51]. In summary the researchers developed a wide range of innovative approaches that individually and collectively advanced the state of the art on the topic of bio-inspired membrane networking.

3.0 Methods, Assumptions and Procedures

3.1 Task 2.1: Membrane Types for Protocol Analysis. The objective of this task was to investigate the effects of membrane agent mobility on trust levels and security clearances for agents and resources and to develop dynamic P system models for instantaneous upgrading/downgrading of system and network components [43].

3.1.1 Summary of achieved results: The main objective of this task was to study formal and theoretical characterizations of Distributed P (DP) systems and membrane computing networks in which the mutual trust level among agents and the clearance levels of membrane agents are subject to dynamic mutation. Membrane agent's trust and clearance levels may result from either mobility, due to agents moving across untrusted links or locations, or result from fine-grained security policies for access control. Traditional security mechanisms have limited flexibility to guarantee security properties, as they often are either too weak to safeguard against actual risks, or too stringent, imposing unacceptable burdens on the effectiveness and flexibility of a network infrastructure. A bio-inspired membrane networking approach to solving this problem was to allow for dynamic modification to security policies. In principal every membrane agent can be assigned a different security policy, and also incorporate membrane agents' interactions feedback to the security mechanisms that influence future policies. For instance, access rights can change for a membrane's response to its behavior, and it is possible to precisely model this behavior by observation. In order to assign rights to an agent, its own past history is evaluated at the moment of the request of the service. The model not only allows for dynamic security policies, but also parameterizes the policies on the past history of membranes [56].

The researchers also investigated the feasibility of a Role Based Access Control (RBAC) approach [57]. RBAC is a flexible and policy-neutral access control technology in that it regulates the access of principals to information and system resources on the basis of activities they need to execute in the system. A 'role' identifies a category of membranes, which have all the privileges assigned to the role itself. Each membrane can be a member of several roles in that it may dynamically activate and deactivate upon initialization. Additionally the system can dynamically modify the settings of roles, in order to allow for a dynamic reconfiguration of the security policy. Another way to perform access control is based on mutable trust policies. Mobile components (e.g., hosts and nodes) are classified according to their behavior, and sites have control capabilities which allow them to deny access to those agents whose behavior does not conform to the site's policy. An agent arriving from an untrusted location is deemed untrustworthy and is obliged to be checked with respect to the local policy. Agents that arrive from a trusted location however can forego the security type checking on the grounds that it has been checked somewhere else in the trusted zone and is found to be valid. In this approach trust levels are based on a process trust enriched by a causal reference. Moreover, an agents' reputation is determined not only on the past events, but also prior cause / effect relationships, including interaction with other membrane agents. The concept of a 'reputation system' is intended to be interpreted very broadly, simply meaning any

system where membrane agents record and use information about past behavior of agents, when assessing the risk of future interaction. The framework introduced represents agents' reputation by means of a temporal logic, and includes a model checking algorithm to dynamically verify if an agent fits the policy requirements.

In this task the researchers focused their efforts on modeling trust management systems. Their main objectives were to describe the exact behavior of membrane networks even in the case of different security policies, and to allow membrane agents' interactions to influence future policies and provide feedback to the security mechanisms. The Distributed P system calculus, presented herein, is focused on these two aspects. In the Distributed P (DP) system derivation, a membrane is specified by a pair: a policy α interacting with a protocol P. The policy α informs the protocol P as to what actions are allowed at any given time, and works on the basis of evidence and observations collected from past interactions. At the same time, P interacts with a network of other membranes, and in doing so it produces the observations gathered in α. The protocol P will consult α when making a decision, e.g. whether or not to grant a specific service to a specific principal. Intuitively, this event is represented by the following**: (Policy(α)) (Protocol(P)) and (Network(N)).** Policies were modeled with a decidable "Datalog-like" logic: where a membrane's policy is represented as a set of formulas depending on a set of past observations. Protocols are expressed in a novel Distributed P system "process" calculus, more precisely a variation of a secure distributed P system calculus specifies locations linked by channel names. This can be formalized as $a\{P\}\alpha|\ N$, where *a* is a membrane with protocol P and policy α, in parallel with the rest of the network N. In the Distributed P system model, the action of sending a message to another node is formalized as granting access to a particular resource. Outputs are then guarded by a formula derived from the logic, as represented by $\beta :: a \cdot l\ \tilde{}m$, whereby the corresponding truth value allows the protocol to send $\tilde{}m$ on the channel/method l. Such an idea was then formalized by applying the communication rule (RCom) of the following reduction semantics:

β' α = α' [b · ˜m ˜m] b : ˜mp : ˜x = σa{p ·˜l(˜x).P + P | P }α| b{β' :: a ·˜l ˜m .Q + Q | Q }β' a{Pσ | P }α| b{Q | Q }β.

The membrane *b* can perform the output on ˜l, to the node a, only if its policy β yields β', in this case the communication takes place and the policy α of *a* is modified according to the fact that an input has been received from b. The operations defined on tuples, as the most general unifier returning a substitution σ, whose application in the semantics is conditioned by successful unification. In fact, in order to allow membranes to offer services to generic (as opposed to individually numbered) nodes, a secure DP calculus derivation was introduced to allow a new form of input capability, allowing us to abstract from the communicating membrane. In the rule above, the input is p ·˜l(˜x), where p is a variable, which at the time of interaction is bound to b.

The secure membrane networking model is a powerful tool for expressing several examples of trust-based mobile ad hoc distributed networks. The researchers were successful in casting ingredients from the world of process algebras to address many

interesting theoretical notions in particular behavioral equivalences. The natural separation of membranes into pairs of components, viz. policies and protocols, introduces new notions of equivalences. In particular, besides studying equivalences of membranes, one can focus on protocols as well as policies. The primary technical contribution in this task was to introduce reduction semantics and a theory of observational equivalences for trust-based systems, which captures equivalences of protocols, policies and processes in a single, homogeneous framework. The approach described in [21] is the Role Based Access Control (RBAC), which is a flexible and "policy-neutral" access control technology. It regulates the access of principals to information and system resources on the basis of activities they need to execute in the membrane network system. The essence of RBAC lies with the notion of software based membrane agents, role and permission: membrane agents are authorized to use the permissions assigned to the roles to which they belong. More specifically, RBAC allows for a preliminary assignment of permissions to roles; a membrane may then dynamically activate and deactivate at run time the roles to which the membrane is a member. Generally, the notion of role extends hierarchically so that sub-roles can be identified and the relevant access permissions inherited.

The work completed in task 2.1 was in itself a first step in formalizing this sophisticated framework by equipping the P system calculus with a notion of processes. Each process P is associated with a name r representing the principal and a set ρ recording the roles activated by the principal during the execution, formally r{|P|}ρ. The calculus is completed by two simple primitives to model role activation and deactivation, role R.P and yield R.P respectively. The operational semantics for this extended language necessarily carries a runtime representation of which node is executing a program and which roles this membrane is currently active in. The consequent reduction rules are**:**
r{|roleR.P|}ρ' r{|P|}ρ'{R}and r{|yieldR.P|}ρ' r{|P|}ρ'{R}.
Intuitively, when a process r activates a role R during a session, R must be added to the set of activated roles ρ, for the remainder of session P. Process r will be executed with the updated set ρ and this process is similarly performed for the deactivation of R. The RBAC schemas themselves are represented using a pair of finite relations (U; P) comprised of a mapping of process names and channels to roles and a mapping of roles to performable actions for that role. The performable actions are represented by **R, R!, R''** which represents the ability to activate role R, send on channels of role R! and receive on channels of role R" respectively.

The transitional P system model as described in Part I of the research project complements the dynamics of the transitional P system rules by providing static guarantees that P systems not representing a given RBAC schema are rejected. Moreover, in this section the researchers introduce a labeled transition system to provide a structured operational semantics to membrane computing programs, and to account for the dynamic checks necessary to enforce RBAC policies. Such a labeled transition system yields bisimulation equivalence, which is adequate with respect to a standard defined congruence. Secure membrane types and simulations are used to deal with three meaningful examples: finding the minimal RBAC schema to execute a system, refining a P System to be well typed with respect to a given security schema, whenever possible,

and minimizing the number of processes in a given system without changing the overall behavior. The concept described herein is aimed at classifying mobile components according to their behavior [83], and at empowering sites with control capabilities which allow them to deny access to those membrane agents whose behavior does not conform to a site's policy. Every site of the P system is represented by k[M| P] as an entity named k and structured in two layers: a computing body P, where programs execute their application, and a membrane M, which regulates the interactions between the computing body and the external environment. A guardian membrane implements the policy each site wants to enforce locally. An agent P wishing to enter a site N must be verified by the guardian membrane before it is given a chance to execute in node N. If the preliminary check succeeds, the agent is allowed to execute, otherwise it is rejected. More importantly, the formalism implements a notion of trust among sites. The guardian membrane contains information regarding the trusted sites within the known network. Intuitively, a membrane arriving from an untrusted location is deemed untrustworthy and is obliged to be checked with respect to the local policy. Membranes which arrive from a trusted location, however, can forego this type of security checking on the grounds that it has been checked somewhere else in the trusted zone and was found to be valid. This is expressed by a membrane migration rule of the form**:**
k[Mk| goTl.P | Q] l[Ml| R] ' k[Mk| Q] l[Ml| P | R]if MlkTP

The relevant parts of the above formulation are P, the agent wishing to migrate from k to l, and l, the receiving site, which needs to be satisfied that P's behavior complies with its policy. The membrane Ml expresses P's policy. The judgment MlkTP represents l inspecting the incoming code to verify that it upholds Ml. The main point is that l verifies the entire code against Ml only if it does not trust k, the signer of P's certificate T. Otherwise, it suffices for l to match Ml against the digest T carried by P go together with P from k, effectively trusting the site k sending the message. Digests consist of multisets of membranes modeled as distributed P (DP) Systems. Essentially, they act as a static abstract representation of the membrane's behavior. The level of abstraction used for digests is contingent upon the particular policy mechanism adopted. For instance, in the setting of policies as multisets of capabilities, these types themselves simply become multisets of potential resource usages. To check that a digest of a program matches a policy one needs to verify that the digest is included, as a multiset, within the policy. The digests for the policies specified using deterministic finite automata take the form of a restricted language of regular expressions which represent the application's potential resource usage. The validity of the secure membrane networking model was verified by proving subject reduction and security type theorems [79]. Here the researchers incorporated a reputation based trust-management system, where membrane agents maintain information about the past behavior of other membranes. This information was used to guide future trust-based decisions about membrane interaction. In particular, membranes' reputation is based not only on the past events, but also on the cause/effect relationships before them, that is on interaction with other membranes [84]. The concept of reputation in the Distributed P system model is meant to be understood very broadly, simply meaning any system in which membranes record and use information about past behavior of membranes, when assessing the risk of future interaction [85].

The framework introduced by the researchers represents behavioral information in a very concrete form. It is essentially an event-structure framework, based on the fact that the observations about membranes' behavior have structure. An event structure is a tuple; ES = (E, ?, #), consisting of a set E of events, and two binary relations on them representing the interrelation between events. An event may depend on another, in the sense that the first may only occur if the second has already occurred. On the other hand, events may be in conflict, in the sense that one may exclude the occurrence of others. The partial order '?' is the (causal) dependency relation, and the symmetric and irreflexive relation '#' is the conflict relation. The advantage of this concrete representation is that it is possible to verify precise formal properties of past behavior, and the basis for making decision regarding future interaction becomes the verification of a behavioral history with respect to a policy. This enables the definition of reputation systems that provides a form of provable 'security' guarantees, intuitively, of the form: "If membrane p gains access to resource r at time t, then the past behavior of p until time t satisfies the requirement '?r', necessary to access r. " As, in the Distributed P system model, the linear sequences of events can be thought of as models if linear temporal logic (LTL), the policy language to describe '?r', is based on a pure-past variant of LTL. In [22] formal syntax semantics are defined, and several examples are provided to illustrate the expressiveness of the definition. Several existing approaches to history-based access control are encoded. Moreover, the work in [23] demonstrated that it is possible to effectively verify if an interaction history satisfies a policy, (i.e., a formula) '?r' can be model-checked. Moreover, this poses an interesting new problem: how to efficiently (dynamically) re-evaluate policies when interaction histories change, as new information becomes available. It turns out that this problem can be solved very efficiently by using an algorithm based on the technique of dynamic programming, used for runtime verification. In particular, although one is verifying properties of an entire interaction story, one does not need to store the entire story in order to verify a policy; prior interaction can be efficiently summarized with respect to the policy. The researchers refined the algorithm introduced in [24], with lower dynamic complexity, that performs some pre-computation on the policy.

3.1.2 Success criteria and expected vs. obtained results. The researchers developed a secured Distributed P system model that was used to treat relevant examples in the area of adaptive network security [25]. In particular, the specific application to the transitional P system rules were tested and proven to work satisfactorily. Moreover, the researchers included several relevant examples in each of the tasks described above.

3.2 Task 2.2 Core Mobility: The objectives of the task were to: (i) refine and extend the definition of resource access in the presence of mobile membrane agents, (ii) devise associated notions of capabilities and primitives for managing membrane agent capabilities, and (iii) to define security policies to handle capability management.

3.2.1 Summary of achieved results: The work on core mobility has uniformly been directed towards the development of new definitions for membrane mobility and resource management, based on an accurate evaluation of their adequacy as Distributed P system abstractions for: 1) mobile ad hoc distributed networks; 2) location-aware applications;

and 3) on the analysis of the behavioral properties which membrane agents convey on those networks. Moreover, the analysis and characterization of membrane behavioral equivalence has received special emphasis, in recognition of the fact that membrane computing equivalence theories are necessary prerequisites to defining and understanding security properties. The researchers investigated several novel approaches of enhanced membrane networking capabilities, and developed them along four different directions. The researchers developed new forms of membrane mobility, governed by capabilities that, when exercised, require the notification of the name of the moving agent to the hosting environment; the researcher investigated different forms of interaction in Distributed P system calculi in which membrane mobility is the objective and involves communications via membrane agents, and in which communication occurs over located channels.

The researchers extended the transitional P system rules so as to address network mobility. Their objective is to eventually deploy physical resources, such as smartcards, which are inherently secure and cannot be copied or modified. They also studied the application of mobility to membrane behavioral theories measuring the dynamic evolution of the distributed environment in which processes and membrane agents operate. In each of these cases, the researchers conducted an in-depth study of the P system type and security foundations of the corresponding Distributed P system calculi. Specifically, for the mobile ambient calculus studied in [26], the researchers provided an inductive characterization of simulation congruence which they used to demonstrate the improved algebraic theory of our Distributed P (DP) system calculus. For the enhanced calculus the researcher applied a proven simulation to ensure it was sound with respect to weak congruence [59], providing in this way a proof method for establishing contextual equivalence. The researchers also included a bi-similarity relation [54] for the extended Distributed P system calculus and used it to prove some of its characteristic algebraic laws. Finally, the researchers developed a secure behavioral theory referred to as, "Secure Membrane Networking", and studied the importance of migration on the behavior of membrane agents operating in an environment in which they are given selective knowledge of dynamically created resources [37] [51]. In each of these prior works, membrane types play different roles, with the exception of [41], where work on a secure type theory is continuing. In [42] secure types are introduced on the top of the ambient calculus in order to capture and verify classical soundness properties. Besides this novel application, membranes are also used to define and enforce resource access properties as in (the extended version of) [51]. In [44] security types play a major role as they are used to devise resource access policies (by assigning different capabilities to access located resources), to define the behavioral equivalence that captures the effect of resource access policies on membrane behavior, and to constrain the mobility capabilities of network nodes. In [53] the author developed new foundations for the calculus of Boxed Ambients (BA), drawing on previous work on refining the primitives for hierarchical communications of the original version of BA.

The unique research contribution included the extension of the transitional P system rules, referred to as Distributed P (DP) system rules. It addresses the wireless communication and mobility interference that could be observed in Mobile Boxed

Ambients (BA) by resorting to co-capabilities and by providing each node with two distinct channels. A local channel enables the interaction of processes local to the node. An upward channel allows communications with the enclosing context. The protocol for value exchange across boundaries is similar in spirit to that of mobility in Secure Ambients, and requires that explicit (mutual) actions be taken by the two parties involved in the interaction. In addition, the DP rules promote movement co-capabilities to the role of binding constructs that inform nodes of the incoming membrane's location or address (e.g., name). Together with a system of password control which verifies the membrane's credentials, this yields a novel way to learn names dynamically, and provides DP systems with essentially the same expressive power as Boxed Ambients. The primary benefit of the expressive DP system calculus is reflected in its simplicity, whose generality, and effectiveness, again relies on end-to-end security. The researchers developed an expressive version of the P system calculus that shares with the original version part of its syntax and all the design guiding principles, while refining several aspects of its semantics and foundational theory [20]. Distributed P system calculus by definition is parametric on the semantics of remote interaction, thus allowing an easier exploration of the resulting network space [63, 64]. The researchers concentrated on behavioral equivalences and constructed a simulation-based equivalence that is a sound technique with respect to weak congruence. While the technique applied is standard, its application is novel, in that it represents the first such known attempt to provide formal accounts of membrane agent duplication, thereby avoiding agent duplication which simplifies the DP system process calculus substantially. This also eliminated many unintended properties that characterize deterministic P systems. Finally in an expanded version of this work the researchers formalized a secure end-to-end secure network and described how this would work in the presence of a particular pattern of interaction. This in turn was used to define and enforce highly distributed resource access control policies [49]. The researchers demonstrated how the novel Distributed P system calculus for Mobile Resources (MR) could be customized for the design and analysis of distributed networks containing mobile, computing devices that may have nested and mobile resources residing in distributed locations that have different capacity and processing requirements [33].

The extended P system rule set accompanying the Distributed (DP) system provides a formal framework to express and prove properties that may depend on the assumption that such resources are neither able to be copied nor modified as such. MR shares ideas with Mobile Membranes, in that both calculi are equipped with nested, named locations. However, like the transitional P system rules, it is the anonymous contents of locations that are moved in MR, and moved by a process external to the location. Resource movement is asynchronous and based on a three-party interaction. The distinctive feature of MR lies in the purely physical nature of resources, which constrains their mobility to the availability of space at the places where they are received, and makes them inherently unable to be copied or modified. The researchers included a reduction as well as labeled transition semantics and prove a correspondence between simulation congruence and higher-order simulation. The researchers provide examples of the expressiveness of their novel DP system calculus, demonstrate how it puts stricter control on mobility and access to locations, and apply the theory to prove its characteristic properties. The researchers arrived at a bisimulation equivalence for the DP system model (e.g., essentially an

enhanced version of the transitional P system rules) in which network processes may migrate between dynamically created locations, and be granted rights to access the local resources these locations offer to their clients. Of specific relevance to the research covered by this task was the study of the effects on the behavioral equivalence of controlling the migration of membranes agents. The control is obtained as an extension of the transitional P system by the introduction of a new ad hoc location capability. The migration between two locations is only allowed if the suitable migration rights are present in the environment. This affects the ability to perform observations at specific locations, as the observer may be denied access. The researchers demonstrated how membrane actions can be modified to take this into account, and generalize the full-abstraction result to this more advanced scenario [69, 70].

3.2.2 Success criteria and expected vs. obtained results. Recall that for task 2.2 the researchers specified the following checkpoints: (1) Provide capabilities that extend the state-of-the-art in providing for end to end secure networks. For instance, it should be possible to control copying, forwarding, and destroying capabilities, and to express actions such as communication at a distance, crossing boundaries, moving resources, upload/download processes, and so on. (2) The researchers devised a simple notion of distributed P systems capable of expressing relevant forms of resource access control. (3) The researchers also conducted a survey of the most commonly used access control policies, and verified that our formalism can express them. The work the researchers completed in this task sufficiently addressed the above mentioned three checkpoints. Firstly, this was done by introducing a coherent set of core mechanisms for capability-constrained computations in wide-area networks. These mechanisms provide for a fine control of agent mobility, by selectively distributing migration rights either based on (dynamic) type-level capabilities, as in [32], or predicated to the possession of shared passwords to be employed in term-level capabilities as described in [33]. One of the unique contributions of this project is that the researchers identified novel approaches to control the effects of migration and distribution of membranes, as well as control the use of physical resources and their allocation in the presence of capacity constraints at the sites where they are received. When combined and complemented with the work on resource management and access control, detailed in task 2.1, the researchers are confident that the core set of mechanisms they derived will provide a solid and effective foundation for the development and analysis of the access control and security policies that constitute the final objectives of this task. Finally the researchers felt it important to remark on the relevance of the work previously developed on membrane behavioral theories. This work provided a deeper understanding of the importance capability-based constraints exert on network resource usage and mobility, specifically with respect to the observational behavior of the systems of interest.

3.3 Task 2.3 Membrane Mobility: The objectives of this task were to extend secure membrane mobility techniques [51] for resource management and information flow to cope with security in the absence of centralized control and to balance static/dynamic techniques to maximize accuracy and expressive power, and efficiency.

3.3.1 Summary of achieved results: The prevailing focus in this task was the application of policies to allow for mobile membrane capability based access control and to determine the extent to which a Distributed P (DP) system based model can be used to enforce them. Access policies are fundamental to network security and originate in early work on resource usage in networks. The rise of distributed networks has only served to reinforce the importance of security mechanisms. Typically the enforcement of policies relies on in-line monitoring, that is, as a process is in execution, the network environment in which the node is operating is augmented to perform policy violation checks at the point at which any resources are requested. Any requests found to be in violation of the policy will then generate an appropriate response from the environment upon node initialization. Although this approach does yield strong security properties, as the node is effectively sandboxed so as to only be given access to exactly the resources allowed by the policy, there is a price to be paid for this arrangement. Real-time monitoring incurs significant execution time overhead through the conditional checks surrounding each and every resource access. Given that the expected behavior of the node, (the membrane agent is used to ensure that the access policies in place are adhered to), it seemed prudent to seek an alternative approach to enforcement which obviates the need for extensive real-time monitoring. This alternative approach can be found in secure static analysis. Here, all resource accesses can be approximated upon network startup and statically checked against the incumbent policy for violations. There is, as yet, no universal formalism in which security policies are expressed and, consequently, they are varied in nature. The most straightforward approach is one of capability access control in which a policy simply specifies a list of resources to which a given process has access and makes no quantitative or temporal restrictions on these resources. More elaborate examples can be envisioned in which resources may only be accessed in strict sequential order and within certain parameters of the environment. Other approaches decouple the users from resources in the policy specification by inserting an additional layer of abstraction in to the policy framework to account for specification in a dynamically changing user base.

The researchers investigated three different, related, approaches as part of this task. Initially the researchers considered the approach in which a framework for static security analysis of access control methods in distributed networks is used where a migrating node is subject to analysis upon arrival at any site in the network. The nature of this analysis is controlled by a guardian membrane agent that specifies the particular policy in place at the site and, moreover, contains information regarding the trusted sites within the known network. In essence, a node arriving from an untrusted location is deemed untrustworthy and is obliged to be checked with respect to the local policy. A node which arrives from a trusted location however can forego the security checking on the grounds that it has been checked somewhere else in the trusted zone and has been found to be valid. The framework itself does not demand a particular formalism for policy specification and in fact, in [34], other researchers have investigated a series of increasingly complex models for this particular situation. The first is the most straightforward, statically declaring for each resource whether access is allowed or not at this location. The model was then augmented with the ability to count usages of resources so that access of a given resource may be permitted for a bounded number of times before it is denied. The DP system model was then extended to include temporal

information by modeling policies as deterministic finite automata. In this case, membranes, operating in accordance to Distributed P system rules, are used to statically verify that an un-trusted incoming node does indeed behave according to the local policy. The use of deterministic finite automata as policies is also taken up in work-in-progress, as in [35], in which the setting is a concurrent functional language where cooperative threads may collectively satisfy a single policy. The higher-order nature of the functional setting here demands a sophisticated treatment of membrane networking systems for policy control. In particular, the receiving node may be instantiated at any point in the temporal flow of the policy and the P system model must then take into account this event. The second approach to access control the researchers considered was that of Discretionary Access Control, (DAC) [63]. Here, membranes are allowed to be grouped into classes and resource management may be controlled by specifying resource access to named groups rather than as individuals. This approach allows for flexibility in specification for dynamically changing users in the P system and has not been employed previously for work on the application of membrane networks for mobility control [85].

An additional dimension to this work included embedding temporal information, as in the case of ad hoc mobile networks, which is necessary for dynamic access control. Rather than constraining the resources to be accessed in a temporal manner directly, the policy specifications demand a flow between the named groups akin to ownership of resources. For instance, a group named "Client" may obtain a print request resource which can only be passed to a group of printers, who subsequently may only pass this request to spoolers. Any other intermediaries in this flow would be considered a violation of the policy. The temporal constraints considered in this work are reasonably expressive as they allow for recursively specified flows of ownership in the sense described above. Below the researcher provide a formalization of this approach based on Distributed P system rules with groups and a secure type P system model for establishing adherence to such policies. The third approach to policy specification that the researcher considered is a reaction to the increasingly prominent concept of Role Based Access Control (RBAC). This is somewhat akin to the use of groups to classify users but extends this concept by shifting the focus of specification to roles rather than membranes or groups of them. Roles identify set tasks in the network which any membrane agent theoretically could engage in. The DP system model allows for nodes to dynamically activate and deactivate their roles upon network initialization and shutdown respectively. Generally, the notion of role extends hierarchically so that sub-roles may be identified and the relevant access permissions inherited. The researchers provided a first step in formalizing this sophisticated framework by extending the transitional P system rules with the relevant commands for role activation and deactivation and providing a formal operational model for this capability. A static type P system is then presented which tracks the dynamic changes role activity and checks that the role based access control, according to the policy specification, is not violated. Finally, as an interesting addendum to this work, it has become apparent that the introduction of sophisticated inherently secure membrane networks for enforcing access control has led to interesting notions of secure encoding relations in these systems. In particular, in some cases, it is not immediately obvious what an appropriate notion of subtype should be. The work the researchers have undertaken is an innovative approach to defining the end to end secure relationship of

bio-inspired Membrane networks. In particular, if one considers the semantics of a P system to be the set of all rules which may be deduced to have that type, then a natural notion of secure encoding arises by considering one type T to be a subset of another U exactly when the semantics of T is a subset of the semantics of U. This is an intuitive definition of this relationship. The work on the above themes is described in partial detail below. The researchers provide a brief overview of main results as a roadmap for the reader.

3.3.2 Success Criteria and expected vs. obtained results: The formulation of this task was intended to provide a framework for managing access control in distributed ad hoc mobile networks in which nodes may migrate between locations. As each location enforces its own access control policies, for static analysis, the absence of access control violations must be verified per location. This task can be alleviated if the node is checked once and can then be certified to be correct, for whichever locations it visits. Certification is only valuable if the certificates can be trusted. This led to a model in which checking of a node upon arrival at a location and trust plays a key role: Networks are structured into sites and each site is split in to two parts: a computing body and a membrane agent which effectively constitutes a secure communication channel, and which regulates the interactions between the computing body and the external environment. A membrane further is comprised of two parts, a mapping of locations in the network to trust values (representing a local view of the trustworthiness of the surrounding network) and the local access policy. The core of the model here is crystallized in the following migration rule**: k[Mk| goTl.P | Q] l[Ml| R] ? k[Mk| Q]l[Ml| P | R] if MlkTP**.

This rule states that for node P to migrate to l from k, it must be checked against the membrane Ml. The check is expressed as MlkTP which requires that l verifies the code P against the policy component of Ml only if it does not trust k, the signee of P's certificate T. Otherwise, it suffices for l to match Ml against the digest T carried with P from k. These digests are the types of the secure P systems presented in this work. Effectively, they act as a static abstract representation of the nodes behavior. The level of abstraction used for digests is contingent upon the particular policy mechanism adopted. For instance, in the setting of policies as multisets of P system capabilities, these membranes simply become multisets of potential resource usages. To check that a digest of a program matches a policy requires that the digest is included, as a multiset, within that policy. The digests for the policies which are based on deterministic finite automata take the form of a computer automata language of regular expressions.

The Distributed P (DP) model the researchers used to formalize Discretionary Access Control builds upon the transitional P system rules with migrating membranes. It construes membranes as processes and it introduces a new class of P systems whose structure supports the specification of fine-grained mechanisms to govern the transmission of addresses (e.g., names), to bound the (iterated) retransmission of capabilities, and to predicate their use on the inability to pass them to third parties. The expanded DP system has the form G[T ?], where G identifies the authority (the process) in control of the values of that type, T describes the structure of those values, and ? is a delivery policy governing the circulation of such values along the channels of the P

system. To illustrate, the type Job [file descSpool ? Print ? Client] is the type of printer job to be first delivered to the spooler, then passed onto the printer, and only then retransmitted back to clients for notification. The Distributed P system relies on secure typing to help achieve a selective distribution of capabilities, based on the groups in control of the communication channels. Thus, the security type of a spooler channel may be defined as follows, with J the type of jobs as given above: Spool [(J)rwSpooler@(J)r; Client@(J)w]. Communications channels with this type may only be delivered according to the intended access control policy, namely with read and write capabilities at the spooler and client sites respectively. Security type preservation provides the basis for a theorem stating that in well-secured processes all names flow according to the delivery policies specified by their types, and are received at the intended sites with the intended capabilities. The model the researchers used to formalize Role Based Access Control extends the transitional P system rules by adding two simple primitives for role activation and deactivation, role R.P and yield R.P respectively. The operational semantics for this extended language necessarily carries a representation of which process is executing code and which roles in which this principal is currently active. Herein the researchers obtained the following evident reduction rules: **r[roleR.P]}ρ? r{[P]}ρ?{R}and r{[yieldR.P]}ρ? r{[P]}ρ??{R}** where r represents the membrane executing the (de)activate instruction and ρ represents the current set of roles in which r is active. The RBAC schemas themselves are represented using a pair of finite relations (U; P) comprised of a mapping of principal names and channels to roles and a mapping of roles to performable actions for that role. The performable actions are labeled as R, R!, R?, respectively, thereby representing the ability to activate role R, send on channels of role R! and to receive on channels of role R?.

The DP system model described in this work is designed to check for non-violation of migrating nodes with respect to an RBAC scheme and therefore the typing judgments are parameterized on these schemes. In particular, for the scheme (U; P), the top-level judgments of systems (collections of executing membrane agents), takes the form ΓPA where Γ is a secure type environment (a mapping from variables, membranes, and communications channels to role and type information for these) which respects U. This latter notion simply asks that the roles assigned to membrane agents and channel names in Γ correspond to the mappings given in U. An interesting auxiliary type relation is introduced for type checking nodes: Γ; ρPrP states that the node P respects Γ and P when it is located in a membrane r with roles ρ activated. Thus ρ acts something like an effect in that it statically tracks the dynamic changes in the active roles. The researchers ensured the validity of the Distributed P system models presented by proving subject reduction and security theorem proofs. Simply put, these ensure that secured programs remain secured throughout their execution and moreover, that secured programs do not contain any policy violations.

3.4 Task 2.4: Network Aware Membrane Systems: The objectives of this task included membrane creation, interaction, and resource access policies applied with respect to distributed mobile ad hoc networks.

3.4.1 Summary of achieved results: Highly distributed networks have now become a common platform for large scale distributed ad hoc mobile networks. While a number of useful internet applications can be developed using the standard client-server paradigm, internet applications distinguish themselves from traditional applications on scalability (huge number of users and nodes), connectivity (both availability and bandwidth), heterogeneity (operating systems and application software) and autonomy (of network nodes and administration domains having strong control of their resources). Hence, other programming paradigms (thin client and application servers, collaborative "peer-to-peer", on- demand, mobile agents) seem more appropriate for applications over internet. These emerging programming paradigms require mechanisms to support mobility of membranes and communications, and effective infrastructures to support coordination and control of dynamic ad hoc networks. For example, a framework to formalize the model of computation of internet applications is clearly needed, and missing. Such semantic framework may provide the formal basis to discuss and motivate controversial design and implementation issues and to state and certify properties in a rigorous way.

More specifically, the researchers addressed crucial issues that needed to be addressed when developing distributed applications include controlling component interactions, since some components can be dynamically downloaded from the network. For instance, the security architectures monitor the execution of mobile nodes to protect a host from external attacks on private information. Recently, the possibility of considering such issues at the level of language design has been explored, aiming at embedding dynamic linking and protection mechanisms in the languages. For instance, the Java language exploits type information and dynamic type checking as a foundation of its security: well-typed Java programs (and the corresponding verified byte code) will never compromise the integrity of certain data. Coordination is a key concept for modeling and designing heterogeneous, distributed, open ended systems. It applies typically to systems consisting of a large number of software components, independently programmed in different programming languages, which may change their configuration during execution. Changes may be due both to external addition/deletion of software components and sites, and to transmission of code and resources made possible by mobility. Complex systems are designed and developed in a structured way, starting from the basic computational components and adding suitable membrane agents referred to as coordinators. The potential reuse of both software based membrane agent coordinators is increased, usually with acceptable overheads. Moreover, the formal description of coordination languages offers a natural framework for stating, checking and proving behavioral properties of open ended systems.

The scenario outlined above is very appealing from the scientific point of view, since many of the concepts to be made precise and most of the open problems involve fundamental issues of computer science research. More specifically, models, languages and logics are needed which are i) distributed, interactive & concurrent; ii) open & reconfigurable, iii) higher order and typed, iv) equipped with abstract compositional semantics, and v) efficiently verifiable. While research has been quite successful on several of these issues separately, their combination is not well understood, and was the object of this study. In graph rewriting systems rules specify local transformations that

can be applied to graphs, possibly concurrently. As such, they are a powerful and flexible formalism suited for the specification of concurrent and distributed systems. They can be considered as a proper generalization of Petri nets, because the states are graphs (instead of multisets), and rewriting rules are able to specify what part of the state must be preserved.

The researchers investigated the use of co-algebras for the specification of interactive systems with a hidden state space as a valid alternative to algebraic methods based on observational equivalences. In this perspective, a natural question is how and to what extent the rich body of techniques and results for the definition and the analysis of transition systems can be transferred to the co-algebraic framework. The theory of mobile and higher order concurrency has been put under scrutiny only rather recently, and DP system calculus has proven to be a trustworthy model. It can be considered as the concurrent counterpart of spi-calculus, which is the basis of most sequential languages. Connections between the two calculi have been studied for sometime in the scientific community. However, only limited results are known about embedding DP-calculus into such efficiently rich higher order systems. Distributed process calculi as applied in DP system models, allowed the researchers to address some key issues of secure distributed networking. By way of example, the researchers investigated the Join-Calculus [34], but several others have also been proposed (e.g. the Secure ss-Calculus, the Distributed ss-Calculus, the Ambient Calculus, and the Seal Calculus). Recent works have studied powerful type systems to enforce security property, e.g. [35], and the approach based on proof-carrying code [36]. This variety of models has motivated the introduction of more general approaches, like Action Calculi and Control Structures [37], Interaction Categories, Rewriting Logic [38], and Tile Logic [39]. All of them abstract at some extent from the concrete syntactical representations of process calculi to develop algebraic, universal models of distributed computing.

Among programming languages, there are languages designed to support distributed scope and access, and languages which provide mobility primitives within a distributed environment. Efficient algorithms and practical verification techniques have been developed for finite state labeled transition systems (automata). Finite state verification has been very successful in all the situations where the control part and the data part can be cleanly separated and the control part is quite complex (e.g. communication protocols and hardware components). Finite state verification techniques are the only case of verification widely used in practice. Finally, the researchers addressed the work on transitional P systems, where the algebraic structure of states is extended, using suitable categorical constructions, to transitions and computations.

3.4.2 Success Criteria and expected vs. obtained results. The general area outlined in the previous section has been specifically instantiated in several original lines of research. The Distributed P system calculus is seen as a bridge from process algebras to mobile and higher order concurrent computing, and studied in detail from several points of view: A concurrent and distributed version of it was developed using graph rewriting techniques, its operational definition was expressed in a restricted format, which automatically yields a reduction semantics; and finite state verification was made

possible for its final version. The researchers have already taken the initial step from the transitional P system rules towards a programming language for mobility and coordination with the definition and initial implementation of the language. Special attention was given to types for security and access control. The approach the researchers devised for the finite state verification of the Distributed P system calculus turned out to be flexible and general enough to handle various history-dependent formalisms, where new actions are declared in certain transitions and referred to later (e.g. process algebras with causality and locality information, Petri nets equipped with history preserving simulation, and DP system-calculus.) Correspondingly, to make the approach independent from the actual syntax, certain classes of computer automata, called History Dependent Automata (HD-automata) have been introduced [40], which are able to allocate and collect names. Without the latter, even simple agents could generate infinite state systems. The researchers are interested in developing an extended format able to represent a hardware based verification environment, and a tool able to port the P system software design to an embedded Application Specific Integrated Circuit (ASIC) design.

In the area of graph writing, the emphasis has been on generalizing analogous results for Petri nets. This work helped the researchers develop a concurrent semantics based on event structures, processes, and unfolding constructions. The researchers also took advantage of this semantics to define the concurrent semantics of other formalisms, like the DP system calculus (already mentioned), concurrent constraint programming, and the synchronous/asynchronous coordination of concurrent processes communicating via shared ports. The researchers initial investigation on simulating transitional P systems evolved into the new model they named Distributed P systems. The basic algebraic structure involved is that of a double category. Here the researchers adapted tiles (i.e. double cells) which consist of rules defining the behavior of open configurations, i.e. system components, which may interact through their interfaces. However, synchronizations and observations are not possible in rewriting logic. Moreover, membrane computing logic handles only closed terms and action sequences, while tile logic handles open configurations and their observations can be multidimensional (e.g., graphs and locality or interaction diagrams.) Thus tile logic proposes itself as a natural, rather general logic for open, distributed, interactive systems with membranes. It supports reasoning about the interactions between the systems being described and their environment.

Ongoing work on tile logic can be divided into three parts. The first is about the foundations of the model itself, and includes work on algebraic and logical properties of double categories and on the definition of first order [41] and higher order [42] versions of them. The second part addresses how useful tiles area as a foundation for open, distributed and interactive systems [43]. Especially interesting are applications to software architectures [44], where the flexibility and compositionality aspects of the tile model can be valuable. The third part concerns how to implement tile logic: a convenient way is to translate it into rewriting logic. The resulting implementation is quite efficient, and the translation has been tested and found acceptable for Membrane Computing models. The theory underlying the translation approach is based on a new link between

single categories and two dimensional categories. Of special interest is the study of data structures to be used for representing configurations and observations [45].

The final area of research interest is with respect to applying co-algebras. Specifically, the researchers are interested in exploring how co-algebraic techniques can be used for representing distributed networks. The collection of states which a co-algebra must form is an algebra with respect to suitable operators (parallel composition). This in turn leads to natural questions about how the static (algebraic) and dynamic (co-algebraic) components of the P systems interact with each other. Preliminary studies [46] indicate that co-algebras defined on a category of algebras are adequate for certain classes of specifications (which include and nontrivially generalize most formats used in process algebras). This setting guarantees that bisimulation is in congruence with respect to the algebraic operators. Another study shows the adequacy of co-algebraic methods to represent mobility [47].

3.5 Task 2.5: Mobility, Control, and Cryptographic Primitives: In this task the researchers devised the optimum way to design mobile ad hoc networks that compute with data sources controlled by multiple nodes, while ensuring compliance with the security policies of the membranes involved. The objective of this task was to devise methods for building mobile secure applications for an information grid consisting of multiple data sources controlled by multiple membrane agents. This process was achieved by employing proven techniques from mathematical logic, P system rules, and logic reasoning to ensure security of applications. This was accomplished while at the same time permitting convenient expression of complex computations with data sources distributed among many nodes operating in a distributed mobile ad hoc network. In order to accomplish this objective the researchers applied automated reasoning tools such as theorem proofs and logical frameworks to prove formally and rigorously the security properties of the Distributed P system model. The intellectual merit of the task consisted of applying scientific and engineering techniques for designing advanced mobile ad hoc networks for computing with multiple data sources that are inherently secure. Mobile network security requires that the trust relationships, access control and information flow policies, and proofs of compliance with these policies are designed in the framework through the use of formal logical methods. These properties were then verified against precise specifications written in a procedure of authorization and information flow using a logical framework so that there is a direct link between the theoretical analysis and the resulting DP system model. This ensured that mobile nodes are in compliance with the security policies of the membranes on the information grid to an extent not previously achievable in conventional networks. This task built upon a secure information grid and associated applications to demonstrate the effectiveness of its approach and provide a means for comparison with competing methods. The broader impacts of this task included the development of fundamental technology to ensure privacy while permitting flexible access to distributed and independently controlled data sources. Membrane security policies and proofs of application compliance with them, readily available is a technical cornerstone for ensuring privacy without unduly limiting the legitimate use of these data sources.

3.5.1 Summary of achieved results. Managers of information repositories today face a tension between security and accessibility of the information they control. Ideally, a network manager wishes to make some set of information readily and conveniently accessible for legitimate uses, while still ensuring that no information is leaked or modified inappropriately or without authorization. However, present-day solutions provide facilities only for one side or the other. On the security side, it is fairly well understood how to prevent unauthorized leakage or modification of information by limiting data accesses to properly privileged queries. This works well provided the gate keeping and authentication protocols are sound and correctly implemented. Yet, even when all works correctly, access is typically limited to a pre-specified set of queries. Moreover, since protocols and implementations often are not correct, it makes good sense to protect particularly sensitive information by keeping it off the network entirely. On the accessibility side, there has been considerable progress in the internetworking community on the problem of ad hoc mobile nodes, wherein mobile ad hoc nodes are able to move across the network to obtain their data of interest. This paradigm allows for considerable flexibility in the exploitation of information, because participants may supply their own applications to be run by other sites. However, such network designs have typically assumed that all participants trust each other. What security is provided focuses mainly on authentication and simple access control, providing no assurance that the accessed information is used appropriately. Research on secure networking has traditionally sought to weaken these assumptions of mutual trust. However, prior work in certified node theory primarily focused on preventing a mobile node from attacking its host or bypassing its host's access controls, not on complex reasoning about authorization or controlling the propagation of information. This tension between security and accessibility, together with the added problem of correctness, has led to an unsatisfying state of affairs. Information is either (1) accessible but insecure, (2) believed to be secure but not very accessible, or (3) definitely secure but completely off-line. The researchers constructed the theoretical and engineering basis for an inherently secure mobile network. In our framework, a node will be free to move throughout the network, but before such a node may execute any application, it must establish to its host that it complies with the host's policy regarding use and propagation of its data.

3.5.2 Success criteria and expected vs. obtained results. The logic described in the previous section allowed the researchers to apply distributed network applications to specify rich information-flow policies. Unlike authorization policies, which can be enforced by the real-time reference monitor, information-flow policies must be enforced statically, this is because information-flows can arise because of what the system did not do, and therefore are a property of the set of all possible system behaviors. It then follows that enforcing information-flow policies depends on being able to statically analyze distributed system software. There is an impressive body of work developing these language-based techniques [48] but in the context of the secure membrane infrastructure there are a number of new challenges. Existing approaches typically rely on membrane networking systems enhanced with rather simple security classification labels that restrict information-flows in the program. For example, in the following code fragment, variables x and y are both integers, but x has label L while y has label M:int{L} x; int{M} y; x = 2 * y; the assignment x = 2 * y is permissible only if the label L is at least as restrictive as

the label M, written ML. Intuitively, data with label L must be treated more carefully than data with label M. If the constraint ML does not hold, the membrane agent should reject this program as insecure. The challenge in our context is to give a logical interpretation to these classification labels that is compatible with both the known program analysis techniques and the policy logic described above.

Decentralized labels permit data owners to specify sets of readers of the data. For example, the label written as {K : R1, . . . , Rn} indicates that K is an owner of the data and K's policy is that R1, . . . , Rn are permitted to read the data. A piece of data might have several owners, each with their own reader constraints. Such labels can be used to implement confidentially policies; data integrity requires additional information to be recorded in the labels. Logically, such a label can simply be interpreted as a proposition of authorization regarding operations on the variable. For example, the label {K : R1, . . . , Rn} on value x can be translated to a conjunction of primitive propositions owns(K, x) ? K affirms (may access (R1, x) ? . . . ? may-access (Rn, x)). In addition, the researcher would have general policies such as the following two: (?K. owns(K, x) ? K affirms may-access (L, x), ? may access (L, x)?K. owns (K, x) ? K affirms may-access (K, x). The first policy permits L to access x if all of the owners permit L to access x; the second says that if K owns x then K can access it. Checking a label condition like ML in the example corresponds to proving a logical entailment M ? L: for every membrane that contains x, the user would have to show that they have permission to read y; otherwise an impermissible information flow is created. The ideas outlined above were extended to a logical interpretation for richer DP system models, such as those for integrity policies that govern write operations to variables [49]. The logical approach also extends naturally to policies that allow delegation relations among membranes, as well as robust declassification mechanisms [50, 51], both of which are useful for constructing practical information-flow policies.

A second challenge was to provide DP system constructs that allow membrane agents to query, react to, and perhaps even change the dynamic authorization policies present at hosts in the distributed network. For example, a policy query might take the following form within the DP system rule base: if (dynamically authorized (may access (L, x))) then P else Q, where, in the node block P, thus one can assume that membrane agent L may access resource x, while code in Q cannot make that assumption. What makes this particularly challenging is that, for such policy queries to be useful, the static analysis of the network model must be able to take into account the results of the query. Thus, dynamically authorized processes cannot consist of a simple routine: it interacts with security type checking in nontrivial ways. There is a considerable design space here that trades off flexibility of the transitional P system rules with feasibility of security type checking. For example, one issue is how initial classifications address the query arguments, for example (L, x), should access be allowed. Another question that needed to be addressed is whether nodes should themselves be able to issue new policy statements, and, if so, how to represent and control this dynamic variation of authority.

4.0 Results and Discussion

4.1 Secure Mobile Network Architecture: There are several key components necessary to allow for secure mobile networking. First and foremost it requires a rich policy language, expressive enough to specify both authorization and information-flow policies. Together, these policies regulate the use and propagation of information throughout the network. During development, it is used to express constraints on allowable mobile node behaviors that can be checked statically. During deployment, policies appear in node certificates which are verified by hosts to ensure compliance with their own local data policies. Finally, during execution of software applications, the policy language provides the vocabulary for authorization checks that are performed at run time to enforce access control. The Distributed P system rule base, described in greater detail below, is based upon open-ended authorization logic. The Distributed P system uses policy rules for specification of information-flow and authorization policies. The secure membrane networking architecture provides features for describing the locality of data sources and the security policies that govern them. This information is expressed as a collection of rules in the Distributed P system language. The secure membrane networking architecture provides three services. First, it checks the certificate accompanying a node it is asked to execute. This protects the host against malicious or corrupted code by ruling out potential flaws (e.g., buffer overflows, etc.) and ensures that the node complies with the host's local information-flow policy. This part of the security enforcement occurs before the node is allowed to enter the network. The certificate verifier is part of the trusted networking base. Second, the membrane networking system manages digital certificates that represent proof witnesses for the authorization and checks the node to make certain it has been authenticated. And third, the membrane networking system provides secure inter-host communication. When, during the course of execution, a program needs to exchange data with or send code to another host in the network, it does so through the membrane networking system, which applies appropriate authentication and encryption to ensure that the underlying communication channel is secure.

4.2 Secure Membrane Networking: The unifying theme of the membrane networking model is its inherent end-to-end security. All the steps in the chain of reasoning showing that a given application program may safely be executed by a given host-including both security policies (including trust relationships among nodes and hosts) and the arguments showing that the program obeys them, are made explicit with concrete evidence in the form of checkable proofs. Membrane network security has a number of appealing advantages. Making the network secure and ensuring policy compliance of the ad hoc mobile nodes protects the hosts of the network from malicious attack. Membrane network security also facilitates auditing the behavior of the system. At run-time, explicitly constructed proofs of authorization decisions make it possible to account for the behavior of the system and, when things go wrong (hosts crashing, passwords or keys being compromised, etc.), to identify and localize the problem. Statically, making the policies secure means that it is possible to check them for consistency and to identify assumptions. Verification of these properties leads to a high degree of confidence in the system's correctness. To summarize, the central idea of this task was making the security of the information infrastructure inherently secure. The soundness of the model was made

manifest by a proof carried out explicitly in a formalized “metalogic”. This was verified by checking that a particular node, in accordance with the DP system, is made manifest through the process of applying security checking to certify completeness. For example, the determination of which hosts are allowed to manipulate what data is made secure through the application of policies that explicitly address the trust relationships among hosts and nodes operating in the network. The authorization decisions initiated by accompanying membrane agent programs are made secure by requiring the DP system to be able to dynamically produce certificates that correspond to proofs in the authorization logic. The policies themselves are made secure as formal objects referred to by these proofs. The outcome of all of this is that it was possible to obtain strong assurance that the DP system model is sound, that the security policies used in a mobile ad hoc network are consistent and consistently enforced, and that any information flows or access control decisions made by the membrane agents comply with the policy.

The researchers also expanded upon existing results and applied standard techniques such as the use of public-key cryptography. The primary assumptions, limitations, and explicit goals of this task were as follows. First, it was assumed that there exists an underlying infrastructure suitable for reliably exchanging data among the hosts of the distributed network (for example, as provided by the standard TCP/IP protocols). Hosts participating in a distributed network, and, in particular, their networking systems, may be trusted to varying degrees by the principals involved in the system (these trust relationships are specified by explicit policies); however the network itself is not trusted. Second, the researchers applied standard cryptographic techniques to ensure confidential and authenticated communication between hosts in the network. For the purposes of model verification, the researchers made the standard assumptions [52] and treat encryption operations as perfect. Third, for the sake of practicality and tractability, some parts of the implementation behavior weren't modeled. For example, low-level details about caching and timing effects were omitted, and the researchers assumed that it is intractable for the attacker to perform complete network traffic analysis. Consequently, there is the possibility that an attacker able to interact with the information grid at a level of abstraction lower than the one modeled may be able to circumvent its information-flow policies. Such abstraction-violation attacks are always possible, regardless of where the abstraction boundary is drawn. Existing work on preventing low-level timing and network traffic analysis attacks could in principle be applied in this context, but it was not the focus in this project. Fourth, this work did address the issues of fault tolerance and reliability that are typically addressed by replication and the use of consensus protocols (although it is likely that those techniques could be fruitfully applied in a distributed network). The work undertaken did not address denial of service attacks, but rather the focus of this work was on integrity of membrane agent software and the confidentiality of data, not on availability of the system. This list is certainly incomplete, but the researchers believe that the philosophy of mobile security will itself help to identify and articulate additional assumptions and limitations that may arise.

4.3 Membrane Security Policies: The researchers applied a logical understanding of authorization and knowledge to reason about and, to enforce properties of infrastructure and of mobile ad hoc nodes communicating in a distributed network [53]. As such it is

mandatory to be able to establish security policies. The first component of this task was the development of an appropriate policy language for authorization and information flow. Since the goal was to design and implement a flexible, open-ended architecture, the specification language itself must be both expressive and open-ended. Moreover, the researchers desired to reason formally about properties of security policies in order to avoid unintended consequences of policy decisions. Finally, the researchers were able to verify membrane agent behavior against these security policies.

This section sketches some underlying logical principles for the DP system policy language and some preliminary evidence for the viability of our approach. The researchers began by decomposing the problem into authorization and information flow. Authorization answers the question of which principals (e.g., nodes) are permitted to access which resources. Information flow specifies the permissible consequences of properly authorized access. Our objective was to identify a logic in which one can reason about whether a principal node should have access to a given resource. However, prior work does not completely satisfy the design criteria-in particular, generality and extensibility is difficult to combine with the ability to reason about policies as a whole (see the related work section 4.4 below). Briefly, a node K should be granted access to a resource R exactly if there is a proof of may-access (K, R). One may understand the meaning of this proposition by considering the pertinent judgments and proof rules. The most basic judgment is that of the truth of a proposition, written as A true. Furthermore what was required was a judgment of affirmation, written K affirms A, expressing a policy of K. For example, K affirms may-access (L, R) is a policy statement by node K that L may access resource R. This implies the truth of may-access (L, R) if K also controls the resource R. The final ingredient is the standard notion of hypothetical judgment. This is written as, Γ =? A true and Γ =? K affirms A, where Γ is a collection of assumptions of the form B true or K affirms B.

The researchers applied the Distributed P system calculus for reasoning about secure authorization as derived in prior research on mobile security [51]. The researchers began with the “so-called” judgmental rules which explicate the meaning of the judgments: Γ, P true =? P true Γ =? A true Γ =? K affirms A. The first rule expresses that from the assumption P one can obtain the conclusion P. The second, that when A is true any node K is prepared to affirm A. Since A is true and has an explicit proof, there is no reason for K to deny it. Conversely, if K affirms A, then A is true from K’s point of view (i.e., one may assume that A is true while establishing an affirmation for the same principal) K:Γ, A true =? K affirms CΓ, K affirms A =? K affirms C. In order to use affirmations within propositions (to form policies that require the conjunction of two affirmations, for example), the logic must internalize them as propositions. The syntax K A packages an affirmation judgment as a proposition: Γ =? K affirms AΓ =? K A true Γ, A true =? K affirms CΓ, K A true =? K affirms C. An authorization policy is just a set of assumptions Γ. An authorization query is a conclusion usually of the form K affirms may-access (L, R) where K controls resource R. Node L will be granted access if there is a proof of the query from Γ. The authorization architecture will apply proof-carrying authorization, wherein L supplies such a proof explicitly for validation by a source monitor embedded in the DP system model. At the root of these proofs are digitally signed certificates that

witness the policy statements of the principals as collected in Γ. This raised several issues, such as how to concretely express policies and proofs, how to assemble proofs, and how to verify their correctness. The DP system model applies a logical framework that is explicitly designed for the representation of logics and proofs. The ensuing design makes the architecture inherently open-ended: thereby making it is possible to enrich our logic with further connectives while still using the same implementation. Furthermore, one can formally reason about the logic and about specific policies using the meta-theoretic reasoning capabilities of the framework. This is useful to establish that a security policy is in fact consistent.

4.4 Information-flow policies: Authorization policies govern which nodes are allowed to access which resources, but they do not specify what those nodes may do with the data once they have permission to access it. Information-flow policies, in contrast, restrict the propagation and dissemination of information throughout the network. Suppose that node L has been granted access to file R (by presenting a proof of access (L, R) to the real-time reference monitor). For example the researchers needed to determine what kind of information flow does this actually entail (i.e., what knowledge can various principles derive). In order to define a logic of knowledge, one needs a new judgment, K knows A, where A is a proposition. Clearly, if K knows A, then A should be true. Consequently, any judgment J entailed by A true is entailed by K knows A:Γ, A true =? JΓ, K knows A =? J. The converse is false, and hence is the very essence of secrecy: there are many true propositions that K does not (and should not) know. One can establish that K knows A by showing that K can infer A using only its own knowledge. This can be formalized using the restriction operator Γ|K, which erases from Γ all hypotheses not of the form K knows B. Γ|K=? A true Γ =? K knows A. As before, one can internalize the judgment K knows A as a proposition, written [K]A:Γ =? K knows AΓ =? [K]A true Γ, K knows A =? JΓ, [K] A true =? J. Note at this point the logic can specify and reason about authorization and its information flow consequences. However, the logic the researchers have applied is monotonic, that is during a proof one can establish more affirmations and infer additional knowledge for the principals, but one can never take away knowledge. Consequently, the DP system can not model consumable resources or systems with essential state changes. In order to capture such systems, it is necessary to move to linear logic [54]. Space permits only the briefest sketch of the resulting secure DP system. The researchers distinguished between persistent assumptions (including all the ones made so far) and linear assumptions, which must be used exactly once in a proof. Such assumptions can model either consumable, one-time certificates (for example, the permission to submit a review, but only once) or modifiable data (such as a paper that may be revised). Consumable certificates are linear assumptions K affirms A, while modifiable data are linear assumptions K knows A. The researchers therefore chose to identify secure DP system models that are intended to go, hand-in-hand, with classes of policies that they can enforce, employing a combination of logical and cryptographic techniques. A particularly natural class is that of stratified policies where information flow may depend on authorization, but not vice versa. Stratification allowed the researchers to generate explicit proofs of authorization without directly relying on potentially private knowledge and enables the use of proof-carrying authorization as an enforcement mechanism.

5.0 Conclusions

The major contributions of this project include the logical foundations of authorization and knowledge and may be summarized as follows:

1) The researchers extended previous work in bio-inspired membrane computing based policy logic by incorporating affirmation (for reasoning about authorization), knowledge (for reasoning about information flow), and linearity (for consumable authorities and resources), combining them into a coherent foundation for a secure networking policy specification.
2) Following the philosophy of mobile membrane security [51], the researchers formalized the meta-theoretical properties of the policy logic; including cut set elimination and various forms of policy analysis, such as noninterference. In this logic, non-interference theorems take the form (Γ, K affirms A =? L affirms C) if and only if (Γ =? L affirms C), under various circumstances, for example, when Γ does not mention K in a negative position and K is distinct from L. This means K cannot interfere with authorization for L. Part of the novel contribution herein was connecting this characterization of noninterference to more standard formulations found in the programming language literature.
3) The researchers explored a set of appropriate cryptographic enforcement mechanisms for the linear authorization logic to account for the presence of consumable certificates and resources. The corresponding problem without linearity is relatively well understood: a statement of the form K affirms A is either directly a digital certificate with contents A signed by a key corresponding to node K, or a chain of formal proof steps ultimately relying on such signed certificates.

Related Work: There have been numerous proposals for authorization logics [51, 55, 56, 57]. Many of these have aims and scopes different from this project in that they are often designed to capture and reason about existing mechanisms, rather than based on purely logical principles. As far as the researchers are aware, these previous logics have not been investigated from a theoretic perspective to prove, for example, cut set elimination and the noninterference theorems that follow from them. Moreover, prior research does not integrate reasoning about knowledge or consumable resources. Another line of related work is where the researchers explored the use of authorization logic for policy enforcement via explicit proof objects. This approach was described in a recent paper taking the first steps at exploring the value of linearity to model consumable credentials [58]. However the previous work had been carried out in the framework of classical higher-order logic which is inherently difficult to reason about. In the current project the researchers approach was designed to be both predicative and constructive [59, 60]. There is also prior research on using logics of knowledge to specify information-flow policies, but that work concentrates mainly on explaining the relationships among different policies and does not consider the interaction of information-flow and authorization [61, 62, 63].

6.0 Recommendations for Future Work

Proposed future research is multifaceted. Initially the researchers wish to carry on fundamental studies in the main stream of networking science, and to contribute to the solution of the practical problems outlined in the state-of-the-art section, employing innovative techniques. The work will be theoretical, but also partially experimental. On the fundamental side, the researchers plan to achieve a better integration of methods and results for the concurrency of higher order systems as described in [66]. The researchers believe that the expressive DP system calculus, in its current version, is a powerful unifying tool for the sequential case and can be used to uniformly model: object oriented and functional programming, normalization, type checking, abstract data types, modules, secure typing, logic frameworks and proof checkers. Extensions of some of these results to distributed and parallel processing could provide some important results. Moreover, the researchers believe long-term work concerning co-algebras would provide improved system performance [65]. The link between concepts like coinduction, final co-algebras, and the notions of bisimulation [54] and synchronization tree, have been previously studied in process algebras application to simple languages like C++ [68, 69]. However, it is still far from being well understood whether the DP system framework can be extended to different forms of distribution, compositionality and higher-order concurrency [70].

On the more practical side, the researchers intend to employ the notion of coordination as a fundamental structuring concept for interactive, open, secure network architectures. Even if the idea of coordination has been around for a considerable time [71, 72], its mathematical foundations are partially missing and the researchers plan to study them relying on the formal notions the researchers developed, such as graph rewriting, tile logic and HD-automata. The long term (and ambitious) goal is to provide a semantic-based environment to specify, validate, implement and test internet based applications [76]. The main principles of our approach are (i) processes and their properties are network aware; (ii) membrane agents function as network coordinators and processes for different entities; (iii) providing properties as a network coordination policy. The idea is that whatever features a dynamically evolving system will have, they will depend on the context of the underlying network (i.e. on membrane agent coordinators). The DP system model adheres to these three design principles.

In a slightly different direction, the researchers intend to develop a methodological framework based on graph rewriting and tile logic for accommodating the semantic integration of aspect-specific visual languages [74]. The design process of distributed systems and of software architecture styles is often based on visual modeling techniques. A variety of visual languages were developed in the past few years, such as entity-relationship and class diagrams, state charts, Petri nets, etc. To cope with the heterogeneity of application domains, domain-specific modeling languages have been proposed which integrate several aspect-specific visual techniques, (e.g. Object Oriented modeling languages like UML and its variations), and Petri-net based languages. Often this integration is done informally and only at the level of syntax, leading to ambiguities which make the languages hard to support. Code generation and formal verification are

examples. It is the researchers' contention that the development of specific algebras of graphs and graph rewriting techniques can be applied consistently not only to the static but also to the dynamic nature of system components. Also, tiles naturally integrate graph descriptions at both the static and the dynamic level, e.g. actor systems and interaction diagrams. In addition, tiles can be drawn employing suggestive, wire-and-box diagrams which can be composed in a simplistic fashion. A visual specification language based on tiles would take advantage of these aspects.

The researchers also plan to pursue the following aspects in the context of enforcing information-flow policies: (1) Develop a system design open architecture blueprint wherein information-flow policies are specified in a way compatible with the authorization logic, following the ideas described above. This will require clarifying the connections between P system-based and logical formulations of noninterference properties. (2) Develop techniques to reconcile the static parts of the information-flow policy specified in the program text itself with the dynamic authorization policies that are enforced by the DP system. The connection between static and dynamic policies will take the form of P system constructs (such as the dynamically authorized operation described above) whose static P system rules reflect the results of dynamic checks. With respect to incorporating logic for security within P systems, the researchers designed and implemented a variation of the transitional P system rules by incorporating P system rules capable of enforcing high-level authorization policies described as simple logic programs. The fundamental feature of a distributed network is that it comprises many data sources or repositories distributed across multiple administrative domains. Thus, data sources are inherently located by virtue of being controlled by unrelated principals, each with their own security and privacy policies. It is important to note that the concepts of administrative locality and physical locality are independent notions. For example, the data governed by a node might be physically distributed across many sites to ensure high availability and reliability, yet would constitute a single locale from the point of view of security policy. Conversely, a single physical repository of data might well be divided into many administrative domains controlled by separate nodes. To avoid confusion, the researchers use the term local to refer to an administrative domain, and the term site for a physical location. Although the researchers focused their attention on administrative locality in this project, it also seems possible to model some aspects of physical locality, such as the "untrusted" nature of communication channels between sites, by introducing notional administrative locales with, for example particularly weak security policies [60]. An adequate model for a secure mobile network must take account of the existence of disparate locales in order to ensure compliance with the security policies imposed by the various principals operating within the network [77, 78].

In order to evaluate the flexibility of their design; verify the practicality of their research approach; and to demonstrate a complete application of a secure mobile ad hoc distributed network, the researchers plan to develop a significant hardware implementation component. There are three main components of the planned implementation, including general purpose tools used in creating distributed application systems [70], the real-time infrastructure and certifying membrane agent, and an instance of a specific application. The real-time infrastructure is responsible for three important

tasks: checking the security certificates accompanying a node when it is deployed, managing digital certificates for authorization checks while the membrane agent software executes, and providing secure inter-host communication. To implement the necessary encrypted communication channels the researchers intend to use existing software packages such as OpenSSH. The researchers plan to adapt current technology on the certifying components of security-typed languages [69, 71, 72]. Although this involves substantial work, the researchers believe that, for the most part, existing techniques will be applicable [82]. Finally, the researchers will apply a logical framework to implement a specific distributed network access control calculus [79]. This application has comparatively sophisticated authorization and information-flow requirements, which will allow the researcher to evaluate the practicality and effectiveness of their design techniques [80, 86].

7.0 References:

[1] Gh. Paun: P Systems with Active Membranes: Attacking NP-Complete Problems. Journal of Automata, Languages and Combinatorics, 6, 1 (2001), 75-90.
[2] A. Obtulowicz: Deterministic P Systems for Solving SAT Problem. Romanian Journal of Information Science and Technology. 4, 1-2(2001), 195-202.
[3] A. Paun: On P Systems with Membrane Division. In Unconventional Models of Computation (I. Antoniou, C.S. Calude, M.J. Dinneen, eds.), Springer, London, 2000, 187-201.
[4] S.N. Krishna, R. Rama: A Variant of P Systems with Active Membranes: Solving NP-Complete Problems. Romanian Journal of Information Science and Technology, 2, 4, 1999, 357-367.
[5] M. Ito, C. Martin-Vide, Gh. Paun: A Characterization of Parikh Sets of ETOL Languages in Terms of P Systems. In Words, Semigroups, and Transducers (M. Ito, Gh. Paun, S. Yu, eds.) World Scientific, Singapore, 2001, 239-254.
[6] M. Madhu, K. Krithivasan: P Systems with Membrane Creation: Universality and Efficiency. Proc. Third International Conference on Universal Machines and Computations, Chisinau, Moldova, 2001 (M. Margenstern, Y. Rogozhin, eds.), Lecture Notes in Computer Science, Springer, Berlin, 2001, 276-287.
[7] A. Rodriquez-Paton: On the Universality of P Systems with Membrane Creation. Bulletin of the EATCS, 74 (June 2001), 229-234.
[8] J. Castellanos, Gh. Paun, A. Rodriguez-Paton: P Systems with Worm Objects. IEEE 7'th Int. Conference on String Processing and Information Retrieval, SPIRE 2000, La Coruna, Spain, 64-74.
[9] S.N. Krishna, R. Rama: P Systems with Replicated Rewriting. Journal of Automata Languages, and Combinatorics, 6, 3 (00), 345-350.
[10] C.S. Calude, Gh. Paun, "Computing with Cells and Atoms: An Introduction to Quantum, DNA and Membrane Computing," Taylor and Francis, 2001.
[11] Gh. Paun, "Membrane Computing- An Introduction." Springer, 2002.
[12] D. Das, "Development of a Simulation Model for P systems with Active Membranes" Final Report. Contract #FA 8750-05-2-0042, March 2006. AFRL, Rome, NY.
[13] D. Das and T. Renz, "A Simulation Model for P Systems with Active Membranes", NanoSingapore 2006, IEEE Conference on Emerging Technologies-Nanoelectronics, January 10-13, 2006, Singapore.
[14] S. Amara S.Nature, 1992, 360, 420-421
[15] R. Radian and BL Kanner, Biochemistry, 1983, 22, 1236-1241
[16] G. Rudnick, Journal of Bioenergetic & Biomembranes, 1998, 30, 173-185
[17] J.S. Hong and H.R. Kaback, Proc. Nat. Acad. Science, 1972, U.S. 69, 3336
[18] A. Androutsellis-Theotokis, N.R., Goldberg K. Udea, T. Beppu M.L. Beckman, S. Das, J.A. Javitch & G. Rudnick, 2003, J. Biol. Chem. 278 , 12703-12709
[19] International technology roadmap for semiconductors. Semiconductor Industry Association, retrived from http://public.itrs.net/Files/2001 ITRS, 2001.
[20] H. Abelson, D. Allen, D. Coore, C. Hanson, E. Rauch, G. J. Sussman, and R.Weiss. Amorphous computing. Communications of the ACM, 43(5):74-82, May 2000.
[21] P. J. Ashenden. The Designer's Guide to VHDL. Morgan Kaufmann Publishers,

Inc., San Francisco, CA, 1996.
[22] G. Ciobanu and G. Wenyuan. A parallel implementation of the transition P systems. In A. Alhazov, C. Martin-Vide, and G. Paun, editors, Proceedings of the MolCoNet Workshop on Membrane Computing (WMC2003), volume 28/03, page 169, Tarragona (Spain), 2003. Rovira I Virgili University, Research Group on Mathematical Linguistics.
[23] P. Dittrich, J. Ziegler, and W. Banzhaf. Artificial chemistries: a review. Artificial Life, 7(3):225-275, 2001.
[24] G. Ciobanu and D. Paraschiv. Membrane software. A P system simulator. Fundamental Informaticae, 49(13):61-66, 2002.
[25] P. Fitzgibbons, D. Das, and T. Renz, "Bio-inspired Membrane Networks for Adaptive Network Security", International Congress of Nanotechnology (ICN), October 31- November 4, 2005, San Francisco, CA.
[26] G. Fauconnier and M. Turner. The Way We Think: Conceptual Blending and the Mind's Hidden Complexities. Basic Books, 2002.
[27] S. De Franceschi and L. Kouwenhoven. Electronics and the single atom. Nature, 417:701-702, June 13 2002.
[28] L. Garber and D. Sims. In pursuit of hardware-software codesign. IEEE Computer, 31(6):12{14, June 1998.
[29] C. G. Langton. Self-reproduction in cellular automata. Physica D, 10:135-144, 1984.
[30] M. Madhu, V. S. Murty, and K. Krithivasan. A hardware realization of P systems with carriers. Poster presentation at the Eight International Conference on DNA based Computers, Hokkaido University, Sapporo Campus, Japan, June 10-13 2002.
[31] D. Mange, M. Sipper, A. Stauer, and G. Tempesti. Toward robust integrated circuits: The embryonics approach. Proceedings of the IEEE, 88(4):516-540, April 2000.
[32] N. Mathur. Beyond the silicon roadmap. Nature, 419(6907):573-575, October 10 2002.
[33] C. Teuscher. Amorphous Membrane Blending and Other Unconventional Computing Paradigms. PhD thesis, Swiss Federal Institute of Technology (EPFL), Lausanne, Switerland, 2004. To be published.
[34] C. Teuscher, D. Mange, A. Stauer, and G. Tempesti. Bio-inspired computing tissues: Towards machines that evolve, grow, and learn. BioSystems, 68(2)(3):235-244, February-March 2003.
[35] S. M. Trimberger. Field-Programmable Gate Array Technology. Kluwer Academic Publishers, Boston, 1994.
[36] A. Tyrrell, E. Sanchez, D. Floreano, G. Tempesti, D. Mange, J.-M. Moreno, J. Rosenberg, and Alessandro E. P. Villa. Poetic tissue: An integrated architecture for bio-inspired hardware. In A. M. Tyrrell, P. C. Haddow, and J. Torresen, editors, Evolvable Systems: From Biology to Hardware. Proceedings of the 5th International Conference (ICES2003), volume 2606 of Lecture Notes in Computer Science. Springer-Verlag, Berlin, Heidelberg, 2003.
[37] F. Varela, H. Maturana, and R. Uribe. Autopoiesis: The organization of living systems, its characterization and a model. BioSystems, 5:187-196, 1974.
[38] J. Villasenor and W. H. Mangione-Smith. Configurable computing. Scientific

American, 276(6):54-59, June 1997.
[39] Csuhaj-Varju, E., Nola, A.D., Paun, G. Perez-Jiminez, M.J., Vaszil, G., 2005. Editing configurations of P Systems, submitted.
[40] J. Castellanos, A. Rodriguez-Paton, G. Paun, Computing with membranes: P systems with worm-objects, IEEE 7th International Conference on String Processing and Information Retrieval, SPIRE, La Coruna, Spain, 2000, pp. 64-74
[41] S.N. Krishna, R. Rama, P systems with replicated rewriting, J. Automata, Languages, Combin. 6 (2001) pp. 345-350.
[42] P. Bottoni, A. Labella, C. Martin-Vide, G. Paun, Rewriting P systems with conditional communications, in: W. Brauer, H. Ehrig, I. Karhumaki, A Salomaa (Eds.), Formal and Natural Computing. Essays Dedicated to Gregorz Rozenberg, Lecture Notes in Computer Science 2300. Springer, Berlin. 2002, pp. 325-353.
[43] C. Martin-Vide, V. Mitrana, P systems with valuations, in I. Antoniou, C.S. Calude, M.J. Dinneen (Eds.), Unconventional Models of Computation, Springer, London, 2000. pp. 154-166.
[44] C. Braghin, D. Gorla, and V. Sassone. A distributed calculus for role-basedaccess control. In Proc. of 17th Computer Security Foundations Workshop (CSFW'04), pages 48-60. IEEE Computer Society, 2004.
[45] M. Bugliesi, D. Colazzo, and S. Crafa. Type based discretionary access control. In Proc. 15th Int. Conference on Concurrency Theory (Concur), London, LNCS. Springer, 2004.
[46] G. Castagna, R. De Nicola, and D. Varacca. Semantic sub-typing for the pi-calculus, 2004. Submitted.
[47] D. Gorla, M. Hennessy, and V. Sassone. Security policies as membranes in systems for global computing. In Proc. Workshopon Foundations of Global Ubiquitous Computing (FGUC), London, ENTCS.Springer, 2004.
[48] N. Nguyen and J. Rathke. A typed static analysis for a concurrent policy based resource access control, 2005.
[49] C. Braghin, D. Gorla, and V. Sassone. A distributed calculus for role-based access control. In Proceedings of 17th IEEE Computer Security Foundations Workshop, CSFW'04, pages 48-60. IEEE Press, 2004.
[50] M. Carbone, M. Nielsen, and V. Sassone. A calculus of trust management. In 24th Conference on Foundations of Software Technology and Theoretical Computer Science, FST&TCS 2004, number 3328 in LNCS, pages 161-173, 2004.
[51] D. Gorla, M. Hennessy, and V. Sassone. Information Society Technologies (IST) Programme MyThS Models and Types for Security in Mobile Distributed Systems Contract IST-2001-32617 (MyThS). Retrieved from the following URL: http://www.informatics.sussex.ac.uk/projects/myths/
Security policies as membranes in systems for global computing. In Foundations of Global Ubiquitous Computing, FGUC 2004, ENTCS. Elsevier, 2004.
[52] K. Krukow, M. Nielsen, and V. Sassone. A formal framework for concrete reputation systems with applications to history-based access control.
[53] M. Bugliesi, S. Crafa, M. Merro, and V. Sassone. Communication interference in mobile boxed ambients. In M. Agrawal and A. Seth, editors, FST&TCS 2002, volume 2556 of Lecture Notes in Computer Science, pages 71-84. Springer, December 2002.

[54] G. Castagna and F. Zappa Nardelli. The Seal Calculus revisited: contextual equivalence and bisimilarity. In M. Agrawal and A. Seth, editors, FST&TCS '02, 22th Conference on the Foundations of Software Technology and Theoretical Computer Science, volume 2556 of Lecture Notes in Computer Science. Springer, Kanpur, India, December 2002.
[55] J. Godskesen, T. Hildebrandt, and V. Sassone. A calculus of mobile resources. In CONCUR 2002 (13th. International Conference on Concurrency Theory), volume 2421 of Lecture Notes in Computer Science. Springer, 2002.
[56] M. Hennessy, M. Merro, and J. Rathke. Towards a behavioural theory of access and mobility control in distributed systems. Technical Report 2002:01, COGS, University of Sussex, 2002.
[57] C. Braghin, D. Gorla, and V. Sassone. A distributed calculus for role-based access control. In Proc. of 17th Computer Security Foundations Workshop (CSFW'04), pages 48-60. IEEE Computer Society, 2004.
[58] M. Bugliesi, D. Colazzo, and S. Crafa. Type based discretionary access control. In Proc. 15th Int. Conference on Concurrency Theory (Concur), London, LNCS. Springer, 2004.
[59] G. Castagna, R. De Nicola, and D. Varacca. Semantic sub-typing for the pi-calculus, 2004.
[60] D. Gorla, M. Hennessy, and V. Sassone. Security policies as membranes in systems for global computing. In Proc. Workshopon Foundations of Global Ubiquitous Computing (FGUC), London, ENTCS. Springer, 2004.
[61] N. Nguyen and J. Rathke. A typed static analysis for a concurrent policy based resource access control, 2005.
[62] C. Braghin, D. Gorla, and V. Sassone. A distributed calculus for role-based access control. In Proc. of 17th Computer Security Foundations Workshop(CSFW'04), pages 48-60. IEEE Computer Society, 2004.
[63] M. Bugliesi, D. Colazzo, and S. Crafa. Type based discretionary access control. In Proc. 15th Int. Conference on Concurrency Theory (Concur), London, LNCS. Springer, 2004.
[64] G. Castagna, R. De Nicola, and D. Varacca. Semantic sub-typing for the pi-calculus, 2004.
[65] D. Gorla, M. Hennessy, and V. Sassone. Security policies as membranes in systems for global computing. In Proc. Workshopon Foundations of Global Ubiquitous Computing (FGUC), London, ENTCS.Springer, 2004.
[66] N. Nguyen and J. Rathke. A typed static analysis for a concurrent policy based resource access control, 2005.
[67] M. Abadi and G. Plotkin. A calculus for cryptographic protocols: The spi calculus. Information andComputation, 148, 1999.
[68] M. Abadi. Logic in access control. In Proceedings of the 18th Annual Symposium on Logicin Computer Science (LICS'03), pages 228-233, Ottawa, Canada, June 2003. IEEE ComputerSociety Press.
[69] M. Abadi, A. Banerjee, N. Heintze, and J. Riecke. A core calculus of dependency. In Proc. 26th ACM Symp. on Principles of Programming Languages (POPL), pages147-160, San Antonio, TX, January 1999.

[70] M. Abadi, M. Burrows, B. Lampson, and G. Plotkin. A calculus for accesscontrol in distributed systems. ACM Transactions on Programming Languages and Systems,15(4):706-734, October 1993.
[71] J. Agat. Transforming out timing leaks. In Proc. 27th ACM Symp. on Principles of Programming Languages (POPL), pages 40-53, Boston, MA, January 2000.
[72] A. Ahmed, A. Appel, and R. Virga. A stratified semantics of general references embeddable in higher-order logic. In Proceedings of the 17th Annual IEEE Symposium on Logic in Computer Science (LICS 2002), pages 75-86, Copenhagen, Denmark, July 2002.
[73] S. F. Allen, R. L. Constable, R. Eaton, C. Kreitz, and L. Lorigo. The Nuprl open logical environment. In David McAllester, editor, Automated Deduction-CADE-17: 17th International Conference on Automated Deduction, volume 1831 of Lecture Notes in Artificial Intelligence, pages 170-176. Springer-Verlag, 2000.
[74] A. Appel. Foundational proof-carrying code. In 16th Annual IEEE Symposium on Logic in Computer Science (LICS '01), June 2001.
[75] A. Appel and E. Felten. Proof-carrying authentication. In G. Tsudik, editor, Proceedings of the 6th Conference on Computer and Communications Security, pages 52-62,Singapore, November 1999. ACM Press.
[76] L. Bauer. Access Control for the Web via Proof-Carrying Authorization. PhD thesis, Princeton University, November 2003.
[77] L. Bauer, K. Bowers, F. Pfenning, and M. Reiter. Consumable credentials in logic-based access control. Technical Report CMU-CYLAB-06-002, Carnegie Mellon University, February 2006.
[78] L. Bauer, S. Garriss, J. McCune, M. Reiter, J. Rouse, and P. Rutenbar. Device-enabled authorization in the Grey system. In Proceedings of the 8th Information Security Conference (ISC'05), pages 431-445, Singapore, September 2005. SpringerVerlag LNCS 3650.
[79] L. Bauer, S. Garriss, and M. Reiter. Distributed proving in access-control systems. In V. Paxon and M. Waidner, editors, Proceedings of the 2005 Symposium on Securityand Privacy (S&P'05), pages 81-95, Oakland, California, May 2005. IEEE Computer SocietyPress.
[80] E. Bertino, B. Catania, E. Ferrari, and P. Perlasca. A logical framework for reasoning about access control models. ACM Trans. Information. Systems Security, 6(1):71-127, 2003.1
[81] M. Blaze, J. Feigenbaum, and A. Keromytis. KeyNote: Trust management for public-key infrastructures (position paper). Lecture Notes in Computer Science, 1550:59-63,1999.[16] Tijn Borghuis and Loe M. G. Feijs. A constructive logic for services and information flow in computer networks. The Computer Journal, 43(4):274-289, 2000.
[82] T. Brauner and V. de Paiva. Towards constructive hybrid logic (extended abstract). In Elec.Proc. of Methods for Modalities 3, Nancy, France, September 2003.
[83] L. Cardelli and A. Gordon. Mobile ambients. In Foundations of Software Science and Computation Structures: First International Conference, FOSSACS '98. Springer-Verlag,Berlin Germany, 1998.
[84] L. Cardelli and A. Gordon. Types for mobile ambients. In Symposium on Principles of Programming Languages, pages 79-92, 1999.

[85] R. Chadha, D. Macedonio, and V. Sassone. A distributed Kripke semantics. Technical Report 2004:04, University of Sussex, 2004.
[86] S. Chong and A. Myers. Security policies for downgrading. In Proceedings of the ACM conference on Computer and communications security, pages 198-209, New York, NY, USA, 2004. ACM Press.